Development of Risk-based Maintenance for Marine Vessels

by

Jane Elizabeth Cullum, BE(Hons) Chemical and Materials Engineering

National Centre for Maritime Engineering and Hydrodynamics

Australian Maritime College

Research Training Centre for Naval Design and Manufacturing

Submitted in fulfilment of the requirements for the degree of Doctor of Philosophy

University of Tasmania

March 2019

Declarations

Authority of Access

The publishers of the papers comprising Chapters 1-2 of the present Thesis hold the copyright for their content, and access to content should be sought from the respective journals. The remaining non-published content of this Thesis may be made available for loan and limited copying and communication in accordance with the Copyright Act 1968.

Statement of Ethical Conduct

All research within this Thesis abides by the international and Australian codes regarding human and animal experimentation, the guidelines created by the Australian Government's Office of the Gene Technology Regulator and the rulings of the Safety, Ethics and Institutional Biosafety Committees of the University.

Declaration of Originality

This Thesis does not contain material which has been accepted for a degree or diploma by the University of Tasmania or any other institution, except that which can be considered background or supporting information which is acknowledged accordingly. To the best of my knowledge and belief the content of this Thesis is original, and not previously published or written by another person except where appropriately referenced. To the best of my knowledge the content of this Thesis does not include material which infringes copyright held by other persons.

Signed:

Jane Cullum

Date: 26/03/2019

Statement of Co-Authorship

The following people contributed to the publication of work undertaken as part of this thesis:

Jane Cullum[1], *Candidate*
Associate Professor Jonathan Binns[1], *Primary Supervisor*
Michael Lonsdale[2], *Industry Partner*
Professor Kiril Tenekedjiev[1,3], *Co-Supervisor*
Dr. Rouzbeh Abbassi[4], *Co-Supervisor*
Dr. Vikram Garaniya[1], *Co-Supervisor*
Professor Nataliya Nikolova, [1,3], *Co-Author*

Their affiliations are as follows:

1. National Centre for Maritime Engineering and Hydrodynamics, Australian Maritime College, University of Tasmania
2. Fleet Marine Services Contract, Serco Defence Asia-Pacific
3. Nikola Vaptsarov Naval Academy
4. School of Engineering, Macquarie University Sydney

Author Details and Their Roles:

Paper 1 (Chapter 1):

Cullum, J. and Tenekedjiev, K. (2017).

Implementing Risk-Based Maintenance Scheduling in the Naval Context: Review and Directions.

Mechanics of Machines, 111, 60-66.

Jane Cullum and Kiril Tenekedjiev contributed to the conception, design and critical revision of the study.
Percentage of Contribution: Candidate 75%, K. Tenekedjiev 25%

Paper 2 (Chapter 1):

Cullum, J., Binns, J., Lonsdale, M., Abbassi, R., & Garaniya, V. (2018).

Risk-Based Maintenance Scheduling with application to naval vessels and ships.

Ocean Engineering, 148, 476-48.

Jonathan Binns, Michael Lonsdale, Rouzbeh Abbassi and Vikram Garaniya contributed to the conception, design and critical revision of the study.
Percentage of Contribution: Candidate 80%, J. Binns 5%, M. Lonsdale 5%, R. Abbassi 5%, V. Garaniya 5%.

Paper 3 (Chapter 2):

Cullum, J., Nikolova, N., Tenekedjiev, K. (2019).

Expected utility analysis of infinite compound lotteries.

International Journal of General Systems, *48*(2), 112-138.

Jane Cullum, Nataliya Nikolova and Kiril Tenekedjiev contributed to the conception, design and critical revision of the study.
Percentage of Contribution: Candidate: 40%, N. Nikolova 30%, K. Tenekedjiev 30%.

We the undersigned agree with the above stated proportion of work undertaken for each of the above published peer-reviewed manuscripts contributing to this Thesis.

Signed:

Associate Professor Jonathan Binns

Primary Supervisor
National Centre for Maritime Engineering and Hydrodynamics
Australian Maritime College,
University of Tasmania

Date: 10/04/19

Professor Shuhong Chai

Principal
Australian Maritime College,
University of Tasmania

Date: 11/04/19

Acknowledgments

"We choose to go to the Moon! We choose to go to the Moon...We choose to go to the Moon in this decade and do the other things, not because they are easy, but because they are hard; because that goal will serve to organize and measure the best of our energies and skills, because that challenge is one that we are willing to accept, one we are unwilling to postpone, and one we intend to win, and the others, too." – John F. Kennedy, September 12, 1962

The research contained in this Thesis was a three-year journey of personal and professional development. I would like to express my gratitude and sincere appreciation for those that walked with me.

To my supervisors; Associate Professor Jonathan Binns, Professor Kiril Tenekedjiev, Dr. Rouzbeh Abbassi and Dr. Vikram Garaniya and my industry advisor Michael Lonsdale, thank you for guiding me toward success though intellectual development.

To my academic and industry colleagues from The University of Tasmania, Serco Defence ASPAC and the FSU Condition Monitoring team; thank you for sharing your wisdom and experience.

I would like to thank Serco Defence ASPAC, the Research Training Centre for Naval Design and Manufacturing (RTC-NDM) and the University of Tasmania for enabling my research under the ARC IC140100003 scholarship awarded for my candidature.

A special thankyou to my loving family and family of friends; I appreciate that you were there with me amid joy, despair and change.

I am proud to have made this contribution to science and to have grown as a person, having learned that the ordinary can become extraordinary with enough courage and dignity.

Sincerely,

Jane

Terminology and Abbreviations

Reoccurring terms or abbreviations are tabulated below in alphabetized order.

Term or Abbreviation	Meaning
Alongside	Moored, docked at wharf
CM	Condition Monitoring
DM	Decision Maker
DUE	Direct Utility Elicitation
FCOL	Finite Compound Ordinary Lottery
FFT	Fast Fourier Transform
GN	Gaussian Noise
ICOL	Infinite Compound Ordinary Lottery
LDA	Linear Discriminant Analysis
MAUT	Multi-attribute Utility Theory
MCDM	Multi-Criteria Decision-Making
MD	Main Diagonal (of matrix)
MVA	Multivariate Analysis
OEM	Original Equipment Manufacturer
OD	Off-Diagonal (of matrix)
PM	Preventative Maintenance
Policy	Action or combination of actions with the maximum expected utility calculated from a decision tree
RAN	Royal Australian Navy
RBM	Risk-Based Maintenance
RCM	Reliability-Centred Maintenance

Abstract

Historically, the maintenance of marine vessels has been expensive and time-consuming due to the use of periodic Preventative Maintenance (PM) and Reliability-Centred Maintenance (RCM). Therefore, alternative approaches such as Risk-based Maintenance (RBM) are needed.

The data-driven RBM system described in this Thesis addresses this need by quantifying risk using a supervised classification algorithm and scheduling maintenance using risk-based decision-making. The system incorporates a combination of Machine Learning, Decision Theory and Utility Theory. Decision Theory and Utility Theory can be used to create and evaluate decision trees which incorporate probabilistic elements, known as "lotteries". Accordingly, this enables the system to include all probabilistic information that is disregarded in maintenance decision-making and maintenance scheduling using periodic PM and RCM. The novelties of this approach include: a supervised classification algorithm for risk-quantification; a novel doubt matrix as part of a tool set to interpret the probabilities generated by the classification algorithm; the incorporation of all probabilities simultaneously in risk-based decision-making within decision trees; and the development of a solution method for a special case of Infinite Compound Ordinary Lottery (ICOL). An ICOL is a lottery which is used to represent the infinite series of events that may occur while maintenance is deferred.

The present RBM system is developed for the shipboard Number 2 General Service Pump. The completed maintenance system is used to analyse the CM data collected from the pump and produce a list of maintenance Policies. Additional results showed that the system can also be used to make decisions in ambiguous situations beyond human capacity, considering all probable faults.

With additional data and research this RBM methodology can be expanded to deliver predictive maintenance for a series of pumps, engines, sub-systems, systems and ultimately marine vessels or any mechanical asset.

Table of Contents

Supplementary Material

List of Figures

List of Tables

CHAPTER 1 - Introduction

The maintenance of marine vessels remains an unreasonably expensive exercise (Moore Stephens LLP, 2017), as insufficent innovation has occurred since the 1940s. Meanwhile, maintenance innovation has occurred in other industries such as aviation. This has reduced the average maintenance cost of a military aircraft by a factor of twelve in comparison to a naval vessel, although fleet-level costs are approximately equal (Deputy Assistant Secretary of the Navy, 2019; United States Air Force, 2019). Lower costs in aviation have resulted due to the emphasis placed on Big Data (Burmester et al., 2018) and the role of data in improving maintenance practices and scheduling.

Data-driven maintenance in the aviation industry has been conducted according to the Risk-Based Maintenance (RBM) philosophy (Paté-Cornell & Fischbeck, 1993; Sun et al., 2018), and alternative techniques (Bayoumi et al., 2008; Steadman et al., 2008). RBM ensures maintenance is scheduled only when equipment exhibits a high failure risk. This scheduling approach reduces maintenance cost (Arunraj & Maiti, 2007) and avoids excess maintenance which may introduce additional failure risks such as human error (Islam et al., 2016). Excess maintenance may also reduce the service life of equipment by introducing additional burn-in periods. This is the first point of increased equipment failure before the gradual decrease according to the "bathtub curve" (Klutke et al., 2003).

Despite its potential to improve maintenance and manage cost, RBM has not been discussed with regard to marine vessel applications. Chapter 1 provides this discussion, which is followed by a critical analysis of techniques used in existing RBM studies to determine directions for future applications.

Some of the material in this Chapter has been published by *Mechanics of Machines* and *Ocean Engineering*. The relevant contents of those publications have been edited for inclusion into the Thesis to avoid repetition and to improve readability.

The citations for the relevant publications are:

Cullum, J. and Tenekedjiev, K. (2017).

Implementing Risk-Based Maintenance Scheduling in the Naval Context: Review and Directions.

Mechanics of Machines, 111, 60-66.

Cullum, J., Binns, J., Lonsdale, M., Abbassi, R., & Garaniya, V. (2018).

Risk-Based Maintenance Scheduling with application to naval vessels and ships.

Ocean Engineering, 148, 476-48.

1.1. Background: *The Maintenance of Marine Vessels*

Impeccably timed and performed maintenance ensures that the largest proportion of vessel working hours is achieved throughout a vessel's service life. This is known as availability. Correct timing and efficacy of work also reduces maintenance cost.

The absolute minimum maintenance cost may be achieved through: identifying the optimal time to perform maintenance, considering the maintenance timing changes throughout the lifetime of the equipment (Klutke et al., 2003), the efficacy of maintenance techniques and equipment design. Achieving the absolute minimum maintenance cost is beyond the scope of the present Thesis as consideration is not given to maintenance techniques and equipment design.

On the other hand, incorrectly timed or inadequately performed maintenance reduces availability (Wahid et al., 2018b) and increases maintenance cost as failures occur, or additional failure risks are introduced through excess maintenance. This was described previously with reference to the "bathtub curve" (Klutke et al., 2003).

Ensuring adequate availability while managing maintenance costs has been an area of interest since WWII (Smith, 1989). Research suggests that naval specialists have been struggling since to implement advances in maintenance practise (Wahid et al., 2018a). Costs increase further with an increase in fleet size, vessel complexity and age (Martin et al., 2017). Additional factors are discussed in Section 1.2. Further, some of this cost is due to wasted time and effort as current strategies do not identify the optimal time to perform maintenance, nor consider that maintenance requirements change over time.

Not all vessels are as specialised as those used in Defence applications, so it is not necessary for all commercial shipping organisations to perform the same amount of maintenance. However, current maintenance scheduling practises are so inefficient that commercial organisations still struggle to manage these essential maintenance costs (Eruguz et al., 2017). Contributing factors are discussed in Section 1.2.

Current maintenance strategies used to maintain vessels include Reliability-Centred Maintenance (RCM) and periodic Preventative Maintenance (PM). RCM may incorporate periodic PM.

Over the past 50 years, periodic PM has enabled marine vessels to achieve a sustainable level of availability (Cordle, 2017); prescribing maintenance at fixed intervals set by the Original Equipment Manufacturer (OEM) or using alternative historical failure data. Periodic PM may schedule excess maintenance as its intervals do not account for variations in the optimal time to perform maintenance or the operational profile of the equipment. Excess maintenance may also result in failures caused by external factors such as human error, further reducing equipment availability. Periodic PM also involves an inefficient decision-making approach to select maintenance tasks. Expert-based decision-making and available information are used to order individual tasks and create a maintenance schedule. Thus, the quality of the schedule and efficiency of the maintenance scheduling process depends on both the expert involved as well as the availability of fault information. This can cause the whole process to become subjective and time-consuming.

RCM (Moubray, 1997) determines the maintenance requirements of an application using reliability-based modelling techniques. In comparison to periodic PM, the additional modelling and analysis means that dedicated support and management is required. In system applications, RCM can be used to prioritise the maintenance of equipment using failure rate, lifecycle cost or risk. However, initial values for each piece of equipment can be difficult to estimate in marine applications as commonly, limited historical data has been collected and the equipment can have multiple applications. These disadvantages exist in all data-driven maintenance approaches and must be weighed against their benefits. RCM decision-making is guided using a decision diagram. Individual interpretation of the diagram still introduces some subjectivity into maintenance scheduling and may be time-consuming. It has been suggested that RCM should not be automated (Moubray, 1997) , as guided expert decision-making leads to the best possible outcome.

Excess maintenance due to fixed intervals and time-consuming maintenance scheduling practices contribute to the high maintenance costs associated with sustaining the operation of any marine vessel. As these are inherent characteristics of periodic PM and RCM, the marine industry must investigate alternatives to resolve this long-standing issue.

1.2. State-of-the-art and Problem Definition

As described previously in Section 1.1, current maintenance strategies employed by marine vessels consist of periodic PM and RCM, which are not strictly optimal.

Deficiencies in current strategies, the desirability of improved asset management and advances in technology have not been enough to encourage innovation. There are a greater number of limiting factors preventing the development and adoption of new maintenance strategies.

Maintaining an asset involves carrying out a large number of critical and minor maintenance tasks. There is an inverse relationship between the deferral of minor maintenance tasks and the operational risk involved. A vessel can embark having deferred some or all of the all minor maintenance tasks as the operational risk is at an acceptable level. Deferral of minor tasks is more likely to prevent the operation of aircraft or nuclear power stations as there is a larger level of risk involved in these industries. Maintenance deferral tolerance is the key factor preventing change and improvement (Shorten, 2013) in vessel maintenance scheduling practises. Other factors include the expected return on investment of a new maintenance strategy, access to vessels, training personnel (Cordle, 2017) and poor industry collaboration (Eruguz et al., 2017).

Implementing specialist equipment and a new maintenance strategy aboard a vessel poses further challenges. An organisation may be interested in trialling a new maintenance method aboard one of their vessels, though use of this method requires prior knowledge of all costs involved and its return on investment. Thus, innovation, cost and return on investment are dependent factors which counteract one another and prevent change. Marine organizations must accept justifications for new strategies based initially on studies conducted in other industries, and support innovation by enabling access to vessels and resources.

Prior discussion has emphasised that alternative vessel maintenance strategies must be adopted from other industries. An emerging approach is RBM, successful to date in aviation (Ahmadi et al., 2010; Kumar et al., 1999; Papakostas et al., 2010) and power generation (Khan & Haddara, 2004; Krishnasamy et al., 2005; Yatomi et al., 2004). The RBM framework was developed for the power generation industry (Chen & Toyoda, 1989; Ochiai et al., 2005) to reduce maintenance costs and ensure asset availability. RBM aims to quantify all risks describing the failure of equipment in a Risk Assessment step; and then to adjust maintenance intervals dynamically using the calculated risk in a Maintenance Scheduling step (Arunraj & Maiti, 2007). It is important to emphasise that RBM is *not* a risk mitigation approach. While the methodology assumes that risks are always present, Risk Assessment and Maintenance Scheduling calculations incorporate *all possible risks, such as they are*. RBM is a maintenance scheduling approach only.

Arunraj and Maiti (2007) provide a comprehensive review of RBM with regard to the manufacturing industry, concluding RBM may be used to guide where and when to perform maintenance to achieve an effective use of resources.

1.2.1. Developing RBM for Marine Vessels

Although maritime regulatory bodies are aware of and understand risk-based concepts (Lloyd's Register, 2017; Shorten, 2013), few risk-based investigations have been conducted to date concerning marine vessel maintenance. In existing literature, only Diamantoulaki and Angelides (2013) linked their work with the RBM approach. However, their application was a moored floating breakwater as opposed to a vessel. Eight other studies (Baliwangi et al., 2006; Diamantoulaki & Angelides, 2013; Dinovitzer et al., 1997; Dong & Frangopol, 2015; Giorgio et al., 2015; Handani et al., 2011; Klein Woud et al., 1997; Smith, 1989) emerged as applications of RBM for marine vessels when search terminology was broadened.

In the Risk Assessment step, these studies either assigned a number to represent risk as a Risk Index (Dinovitzer et al., 1997; Klein Woud et al., 1997; Smith, 1989) or estimated failure probabilities (Baliwangi et al., 2006; Diamantoulaki & Angelides, 2013; Dong & Frangopol,

2015; Giorgio et al., 2015; Handani et al., 2011). The former Risk Index method was a more efficent though less accurate approach, while the latter probabilistic approaches provided greater accuracy and consistency, though were less efficient as they required development and interpretation of their results.

Due to the greater accuracy and consistency provided, the development of a probabilistic technique is worthwhile in the Risk Assessment step. However, existing studies cannot guide development as they are application specific, and none of these existing studies considered states other than ‘operational’ or ‘faulty’. Probabilities of all faults are required to select the best possible maintenance action as multiple faults could be occurring simultaneously. Alternative techniques such as Condition Monitoring (CM) and Machine Learning (ML) can be applied to detect individual faults within this context (Gao et al., 2015; Lee et al., 2014).

Expert-based judgement (Diamantoulaki & Angelides, 2013; Dinovitzer et al., 1997; Handani et al., 2011; Klein Woud et al., 1997; Smith, 1989) and optimisation (Baliwangi et al., 2006; Dong & Frangopol, 2015) were used to conduct Maintenance Scheduling in existing studies. Neither of these were ideal. Selection of a maintenance action was only possible using expert-based judgement, though this was a subjective and time-intensive exercise. Conversely, optimisation algorithms could produce maintenance interval solutions quickly and consistently, but not decisively enough to select single actions. A balanced technique is required to ensure decisiveness, speed and consistency in Maintenance Scheduling. Decision Theory (French, 1986) may be applied to achieve this balance, utilising expert knowledge within a probabilistic decision-making system to efficiently produce consistent recommendations. This reduces but does not eliminate subjectivity as expert data is incorporated. The theory does not strictly provide for the calculation of maintenance intervals although intervals can be incorporated into the decision as deferral of maintenance over a specified duration. Ongoing use of such a system should lower maintenance costs and improve availability by increasing decision-making consistency and efficiency.

1.2.2. Developing the RBM Framework

Considering the unique context of marine operations and limitations of existing RBM applications, it was evident that the framework provided by Arunraj and Maiti (2007) required adaptation. A new approach was developed within the present Thesis which schedules maintenance considering that risk is both quantified and considered within the decision-making process. Techniques used to quantify risk are described in Section 2.1.1, while decisions taken under risk are discussed further in Section 2.2.5. The approach also considers that RBM should be compared to other strategies, specifically using availability and overall maintenance cost.

The new framework is shown below in Figure 1-1 as a cycle.

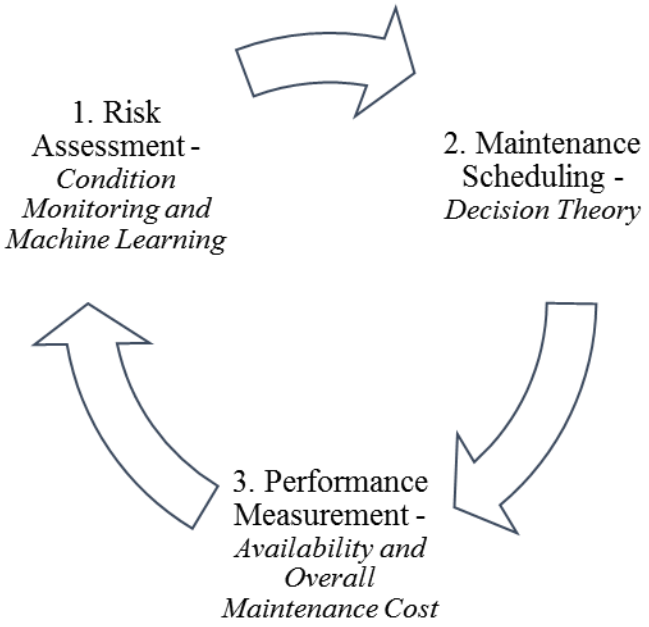


Figure 1-1: Cyclic RBM Framework

Risk Assessment quantifies all risks, Maintenance Scheduling creates a maintenance schedule using the risk and Performance Measurement calculates the availability and overall maintenance cost of the schedule. The results of overall maintenance cost and availability calculations can then be used to reassess the CM data collection frequency for the next Risk Assessment as the cycle repeats. Risk Assessment and Maintenance Scheduling may be considered as separate elements, though are integrated to form the complete system. It is expected that the use of an RBM system

will identify the optimal time to perform maintenance, reduce time spent on maintenance scheduling and generate consistent and cost-effective maintenance schedules. While the present RBM approach was developed considering the marine industry and is applied within this context in the present work, future applications are not limited to this industry. An explanation of the Risk Assessment, Maintenance Scheduling and Performance Measurement elements follows.

1. *Risk Assessment*

This element of the framework is used to assess the condition of the equipment and interpret this information into a probability of each 'no fault found' or 'fault found' state. It is not strictly necessary to view or analyse probabilities generated by this step as they are addressed accordingly by the Maintenance Scheduling element. Suggested techniques for this element include the analysis of CM data in combination with a ML algorithm.

2. *Maintenance Scheduling*

The second element of the framework is used to interpret probabilities of failure into recommended maintenance actions using all available probability information. Comprehension of multiple probabilities of failure simultaneously is not possible for the human expert decision maker. Decision Theory is suggested for this purpose.

3. *Performance Measurement*

The third element of the framework is used to measure the performance of the RBM approach against existing maintenance strategies, such as periodic PM or RCM. Performance measurement is crucial to gain support for further applications and vessel access. This element of comparison was not evident in existing studies. Availability and overall maintenance cost should be used in these evaluations as they are common within the marine industry. For applications outside the marine industry, other relevant measurements should be used.

Some operational data will be available for the most basic availability calculations, however more detailed calculations (Ebeling, 2005) can also be used if data are available. Overall maintenance cost is the sum of the parts, tools and labour necessary for each maintenance task. The cost of

downtime may be included in the labour cost or otherwise incorporated into the overall maintenance cost.

1.2.3. Developing Applications

RBM applications can be divided into a series of levels based on their complexity. The structure in Figure 1-2 can be generalised for applicable classes of vessel or to represent non-marine assets. Irrespective of its size, each future application requires its own RBM system. It is possible to streamline the development of larger applications such as sub-systems and whole vessels if initial work focuses on creating systems for individual pieces of equipment, such as a pump or compressor. The specific RBM system for the pump or compressor can then be linked to other equipment according to its arrangement in the physical system to create a complex application. No component level applications exist to date. A pump application is the focus of the work conducted within the present Thesis. This Number 2 General Service Pump application is outlined further in Section 1.6 and discussed in detail in Chapter 3.

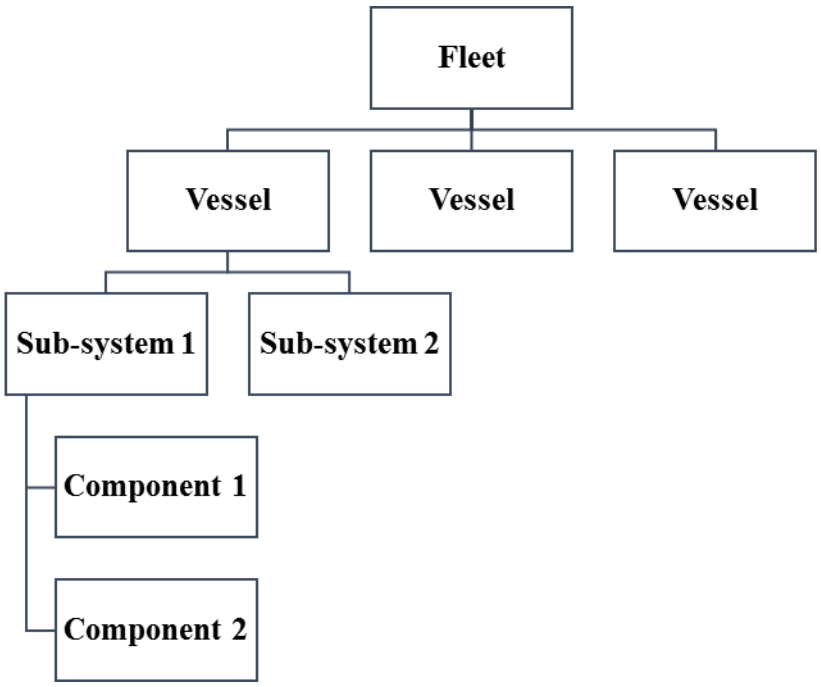


Figure 1-2: Marine Vessel Application Hierarchy

1.2.4. Implementation

A five-phase implementation process is shown in Figure 1-3. This process is applicable irrespective of whether the organisation utilizes periodic PM or RCM.

1.3. Research Directions

The underlying principles of RBM address the current challenges inherent in the maintenance of marine vessels, though RBM is yet to be applied in this field. To assist new RBM applications, Chapter 1 presented: an RBM framework for vessel applications, an application hierarchy and a component-level implementation process.

Several gaps were identified within the previous discussions in this Chapter. The research gaps evident in this field are summarised below:

- Flexibility in maintenance intervals and improved maintenance scheduling practises are required to manage downtime and maintenance cost in marine vessel applications. RBM may improve upon existing practices, but a limited number of applications aboard vessels exist to date and do not include performance measurement.
- An RBM system requires performance measurement to determine its effect on the availability and overall maintenance cost in comparison to an existing maintenance strategy. Availability and overall maintenance cost are suggested for use with marine vessel applications.
- Risk Assessment and Maintenance Scheduling techniques used in existing studies are not general enough to be used across different applications. CM and ML techniques are general techniques which can be applied as a new approach to enable the Risk Assessment element to consider multiple faults. Decision Theory can integrate expert knowledge and data into an automatic decision-making process for improved consistency and efficiency in Maintenance Scheduling. This reduces the subjectivity involved in the process, as decisions are made based on both data and expert knowledge. CM, ML and

Decision Theory can be combined to create an RBM system which identifies the optimal time to perform maintenance considering all possible faults, by making risk-based decisions. The development of this system is the focus of the present Thesis.

- The existing RBM framework, design of applications and their implementation can be adapted for marine vessel applications. An RBM framework for vessel applications was presented in Figure 1-1. A structured approach to guide the development of applications was shown in Figure 1-2. No component-level applications exist to date. The first step in the development of the RBM concept for vessels is a component level application. An RBM system for a pump is developed in the present Thesis. Component-level implementation was shown in Figure 1-3.

I	ESTABLISHMENT	• Identify organisational resources for ongoing management of RBM applications • Identify organisational resources for the development of new RBM applications • Review existing or develop new organisational policy governing the use of RBM Framework

II	APPLICATION DESIGN	• Identification of scope of RBM application
		• RBM development team to procure all relevant information for initial project planning, including:
		  o Operating profile of application
		  o Technical specifications and documentation
		  o Maintenance history of application
		  o List of causes of failure for the given application
		  o List of methods effective in measuring progression of application toward of causes of failure
		  o Experimental data describing occurrences of all causes of failure for the given application, under typical operating conditions
		• Identification of interfacing requirements with existing asset management systems

III	SYSTEM DESIGN AND VALIDATION	• Preliminary design of RBM data management and analysis system - tailoring of Risk Assessment and Maintenance Scheduling techniques for the given application • Preliminary design of RBM system interfaces with asset management systems • *Operational Trial:* Procurement of relevant condition monitoring equipment and acquisition of appropriate CM data • [As required] Procurement of additional experimental equipment, and acquisition of appropriate experimental data • Testing of RBM system and asset management system interfaces • Evaluation of RBM Scheduling application recommendations against existing maintenance regime
IV	DESIGN ACCEPTANCE	• [As required] Rework of RBM system design • Acceptance of final system design • Acceptance of system interface operation
V	INTEGRATION	• Personnel debrief and training • *User Acceptance Trial:* Operate CM equipment and RBM Scheduling system in alignment with organisational policy

Figure 1-3: RBM Scheduling Implementation Approach

1.4. Research Questions

In alignment with the research gaps summarised in Section 1.3, the research questions answered within this Thesis are:

- How should a shipboard component be monitored, and the resulting measurements interpreted to produce condition data?
- How can the specific faults within a shipboard component be identified accurately given condition data?
- How often should maintenance actions occur for a shipboard component to reduce overall maintenance cost and increase availability?
- How can an RBM system for a vessel component be compared to other strategies to demonstrate that further applications are worthwhile?
- How should the system be integrated into daily operations aboard a vessel?
- How can the resulting RBM methodology be improved further, with consideration given to new applications?

1.5. Research Objectives

The research questions in Section 1.4 were developed into the following objectives:

1. Develop an organizational implementation approach for the application of RBM to a piece of equipment.
2. Develop an appropriate ML algorithm for a piece of shipboard equipment to determine its condition.
3. Apply Decision Theory to select the best possible maintenance task for a piece of shipboard equipment.
4. Combine the resulting tools to create an RBM system which translates CM data into a maintenance Policy.

5. Quantify and compare the recommendations of the system using overall maintenance cost and availability with the current maintenance approach over the same period.

1.6. Scope and Limitations of the Study

The scope of the research which is presented subsequently in Chapters 3 to 6 of this Thesis is limited to the development of a maintenance system for a case study pump aboard an assigned vessel. The research aims to determine if the present RBM methodology theoretically improves the availability and reduces the overall maintenance cost of the Number 2 General Service Pump over a 6-month period. The minimum maintenance cost requires further consideration of the effectiveness of maintenance work and equipment design and is beyond the scope of the work conducted in the present Thesis.

The pump application is described in detail in Chapters 3 and 4. The Number 2 General Service Pump was selected as it is a component-level application, which is the first application required for further development of the RBM concept. The selection of the pump also minimises the injury risk during data collection as the working fluid in both the test equipment and the application is water under low pressure. Additionally, the CM characteristics of pumps are well studied, leading to a straightforward data collection regime. Lastly, a component with fewer moving parts requires fewer monitoring parameters, resulting in smaller datasets. This minimises the resources required for CM implementation. Other pieces of equipment may require larger datasets and a greater number of measurement types to characterise their condition.

While a shipboard pump is considered in the present work, the RBM methodology is applicable to any piece of equipment or asset. The methodology only requires that the characteristics of the equipment are measurable and that a suitable expert is involved in system development.

The selection of the pump led to the selection of CM parameters and analysis methods appropriate for horizontal centrifugal pumps. Chapter 2 discusses these parameters and methods.

Experimental and CM investigations were conducted at the commercial premises of the sponsoring industry partner. These investigations and CM aimed to track the development of common failure modes within the pump over a 6-month period.

The availability of financial resources, the vessel and personnel limited the overall quality of the data and therefore the accuracy of the maintenance system. These factors affected the number of experimental trials, duration and frequency of the CM data collection (Chapter 3) and the expected completion timeframe of the study. However, the experimental work in Chapter 3 replicated more faults than considered in similar studies (Kamiel, 2015; Sakthivel et al., 2010). The length of the 6-month monitoring period included two 3-month maintenance cycles, which balanced CM data collection duration and completion timeframe.

Limited historical pump maintenance data prevented the use of more advanced availability or other alternative metrics (Ebeling, 2005).

Four decision contexts were considered due to time and resource limitations, which represented the operational environment of the pump. These included operation of the pump in one of four scenarios: the vessel was alongside and the engine room was quiet, the vessel was alongside and the main engines were running, the vessel was slow steaming (<4 knots) at sea and no emergency occurs and while the vessel was slow steaming at sea and an emergency occurs.

Lastly, the linear discriminant classifiers can be used to develop regression models for condition trend analyses, although were not due to the limited duration of the present work.

1.7. Thesis Organisation and Contributions

The Thesis contributes several novel ideas and discoveries to the body of scientific knowledge surrounding the maintenance of marine vessels. This Thesis consists of six main chapters created from three published journal articles. The beginning of each Chapter lists the relevant publications contained within it. The contents of the present Thesis are outlined below by Chapter or Appendix, concluding with a summary of the overall contribution of the Thesis.

- The previous material in **Chapter 1** provided a critical review of the current maintenance strategies employed for marine vessels, identified that RBM exists for these applications, and introduced a novel RBM framework, application hierarchy and implementation strategy to enhance the applicability of the maintenance framework to marine vessels.
- The first half of **Chapter 2** reviews the literature surrounding the key concepts and existing marine vessel applications of CM and ML. Analysis of CM techniques revealed the general applicability of vibration data in fault detection of marine equipment. The most probable class label outcome is compared to the value of using a classifier to produce all probabilities using three real examples. The discussion highlights the need for the generation and use of all probabilities from classification algorithms in fields such as fault detection of machinery. Probabilities can only be calculated using Bayesian classification algorithms, although approximate quantities can be calculated from non-Bayesian approaches. However, there were no widely used tools which can interpret these probabilities prior to the work in the present Thesis. The largely unknown certainty matrix and the novel doubt matrix are described as interpretation tools for supervised Bayesian classification algorithms. Their approximations are described for non-Bayesian algorithms. These tools are recommended for performance estimation in combination with the confusion matrix. A supervised Bayesian classification algorithm is then described which can be used to generate probabilities. This algorithm creates linear discriminant classifiers using Linear Discriminant Analysis (LDA).
- The second half of **Chapter 2** introduces the theory and concepts behind Decision Theory and Multi-Attribute Utility Theory (MAUT). As maintenance or deferral decisions involve uncertain consequences and are decisions taken under the condition of risk, these uncertain consequences can be modelled as lotteries. When maintenance is deferred, these lotteries can involve outcomes which stretch indefinitely into the future. These lotteries are termed Infinite Compound Ordinary Lotteries (ICOLs). A novel simplification algorithm and solution method are then derived to be able to solve the special recursive case of ICOL. The recursive

case describes a situation following reduction of any ICOL to a maximum reduced ICOL in which the outcomes are the same in number and value over time. Other special cases of ICOLs are addressed in the referenced publication. The Chapter concludes with a summary of its contents.

- **Chapter 3** describes the Number 2 General Service Pump application, followed by the collection and processing of experimental and CM data for LDA. **This Chapter addresses Research Objectives 1 – 3.**
- **Chapter 4** firstly describes how LDA is applied in the Risk Assessment part of the RBM system to create four classifiers. Each represents an operation and maintenance scenario such as 'Vessel at Sea, Emergency'. The confusion, certainty and doubt matrices are then used to estimate the performance of the four classifiers in resubstitution and 'holdout' tests. These are approximate holdout tests as the test data are based on the learning data. The Chapter then describes how Decision Theory is applied to create a corresponding set of four decision trees which complete the maintenance system. Chapter 4 also presents the methods used to measure the performance of the system against another maintenance strategy. A summary then concludes the Chapter. **This Chapter addresses Research Objectives 1 – 3.**
- **Chapter 5** presents the results of the pump RBM system analysis of single-state data, CM data and test data. Considering only one state, the recommendations of the system and an expert are compared. Then, the recommendations of the system are compared with periodic PM for the pump application over the CM period. The system's applicability in ambiguous situations where more than one state is important are discussed, followed by a description of how the present system can be included within a future predictive maintenance system. Lastly, the limitations of the work conducted in the present Thesis are presented. **This Chapter addresses Research Objective 4.**
- **Chapter 6** draws conclusions from the results presented in Chapter 5 and provides directions for future work. **This Chapter addresses Research Objective 4.**

- **Appendix A** presents a derivation of probability formulae discussed in Chapter 2.
- **Appendix B** presents a supplementary maintenance record for the Number 2 General Service Pump.
- **Appendix C** presents all MATLAB software developed for data processing and the maintenance system described within the present Thesis.
- **Appendix D** presents all steps involved data processing within the present Thesis. These are summarised in Section 3.7.
- **Appendix E** presents a sample of raw vibration velocity data from each of the experiments as frequency domain plots, velocity waveforms and acceleration waveforms. Characteristic frequencies are highlighted for each sample.
- **Appendix F** presents the confusion, certainty and doubt matrices resulting from 'holdout' tests using test vectors which were based on the learning data.
- **Appendix G** provides a qualitative description of the consequences used within the lottery models.
- **Appendix H** shows how each consequence attribute value was quantified based on its qualitative description in Appendix G.

The overall Thesis contributes the novel doubt matrix tool, a novel simplification approach and solution method in the case of recursive ICOLs, specific guidance toward developing an RBM system for a given vessel component application, and a unique discussion of its unique capabilities These include: the determination of the optimal maintenance time, and the use of all probability information in consistent and rational maintenance decision-making.

Figure 1-4 illustrates the structure of this Thesis.

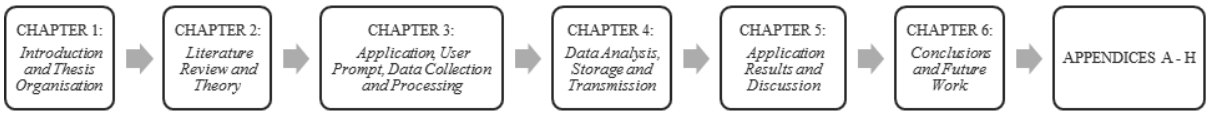


Figure 1-4: Thesis Organisation

1.8. Summary

The introductory sections of this Thesis discussed the current progress toward improving the maintenance of marine vessels. Current strategies were determined to be inefficient uses of time and resources and have the potential to reduce the service life of equipment through excess maintenance work. Innovation is needed to reduce excess maintenance work and maintenance costs. Developing an RBM application was suggested as a key area of research to address the current challenges faced in the maintenance of marine vessels. Developing and evaluating an RBM system is the focus of the research presented in the current Thesis. An RBM system shall be created using CM, ML and Decision Theory concepts applied to the maintenance of a shipboard pump. These concepts are described and evaluated in Chapter 2. Research Objectives and Research Questions were listed and focused on development and validation of the RBM system for a piece of shipboard equipment. The scope of the present study was limited to application of RBM to a Number 2 General Service Pump aboard a vessel, and the limitations of this scope and of the overall study were discussed. Lastly, Section 1.7 outlined the organisation of the Thesis.

CHAPTER 2 - Literature Review:

Developing the RBM Framework

The theory and existing marine vessel applications of Condition Monitoring (CM), Machine Learning (ML) and Decision Theory are discussed within this Chapter. The discussion in Section 2.1 identifies suitable CM and ML techniques which may be included in the development of RBM systems. The present RBM system shall be developed using vibration data alongside other measurements. Appropriate selection of a ML algorithm can be used to manage the cost of an application. Algorithms can be compared using performance estimation tools. ML performance estimation methods are discussed, with emphasis on classification methods which can be used to produce probabilities. The probabilities can be used to fully quantify risk in an RBM system. Previously, there were no widely used performance estimation methods which interpret probabilities. The confusion and certainty matrices are presented and combined with the novel doubt matrix to create a new approach. As probabilities can only be produced using supervised Bayesian classification algorithms, a similar approach is presented for non-Bayesian algorithms. Multivariate Analysis (MVA) is then discussed. MVA shall be applied to create Bayesian linear discriminant classifiers which can produce all state probabilities from equipment measurements. Classifier development for the shipboard pump application is detailed in Chapter 4.

In Section 2.2, existing maintenance scheduling and maintenance decision-making approaches are discussed. None of these consider probabilistic decision-making, which must be used within the RBM system to consider all risks involved in the operation and maintenance of equipment. Decision Theory can be used for risk-based decision-making and shall be applied within the present RBM system to simultaneously consider all probabilities generated by the classifiers in decision-making. This decision analysis for the shipboard pump application is presented in Chapter 4. However, this theory could not previously simplify or solve lotteries which contain outcomes that extend indefinitely into the future. A novel approach is presented which allows the

theory to model maintenance deferral actions which have these indefinite outcomes as Infinite Compound Ordinary Lotteries (ICOLs). A simplification algorithm is presented. A solution method is developed and for the special recursive case and is demonstrated using a quantitative example. Most of the maintenance actions within the RBM system in Chapter 4 are modelled as ICOLs and solved using the simplification algorithm and the recursive solution method. Section 2.3 presents the key conclusions of the present Chapter and is followed by a summary in the final Section 2.4.

Some of the work presented in this chapter has been accepted for publication in the *International Journal of General Systems*. The contents of the paper have been rearranged for better integration into the thesis format. The citation for the relevant article is:

Cullum, J., Nikolova, N., Tenekedjiev, K. (2019).
Expected utility analysis of infinite compound lotteries.
International Journal of General Systems, *48*(2), 112-138.

2.1. Condition Monitoring and Machine Learning for Marine Vessel Maintenance

Data which describe the condition of a piece of equipment are required to develop an RBM system. These data are acquired through CM methods. The presence of faults or the initial symptoms of faults are shown within CM data as changes in equipment behaviour. A 'no fault found' dataset is required for comparisons with 'fault found' data. An ideal CM regime is one which is developed for the specific application as fault symptoms and faults may also be unique to the equipment studied.

ML techniques are a family of algorithms which can be developed to recognise patterns in data. These patterns are limited to trends using regression methods and grouping similar data using classification methods. The interpretation of CM data for maintenance scheduling and predictive maintenance can be conducted using regression methods. The interpretation of CM data for maintenance decision-making can be conducted using classification methods. The application of regression methods is beyond the scope of the work in the present Thesis due to limitations placed on the data collection regime (Refer to Section 1.6). Subsequent discussions refer to ML classification techniques only.

A maintenance system created using a combination of CM and ML techniques could identify the optimum time to perform maintenance. This is achieved through calculation of the most probable class, and the determination of whether this is a 'fault found' class. As discussed previously, performing maintenance at the optimal time reduces maintenance cost, achieves a specified level of availability and maximises the service life of equipment; which is an improvement upon periodic PM and RCM. Such a system can also identify a specific maintenance action to perform if each class is paired with a maintenance action or deferral, under the assumption that the most probable class is certain to occur. Thus, it is reasonable to expect that systems of this type are

used frequently in the maintenance of marine vessels. Section 2.1.1 discusses why this is not the case.

2.1.1. Existing Applications of CM and ML: *Trends and Challenges*

A review of the literature encompassing CM and ML marine vessel applications has identified the following studies summarised in Table 2-1 and Table 2-2.

Table 2-1 and Table 2-2 highlight several trends in the development of maintenance systems which incorporate these techniques. Studies tended to branch out beyond individual components toward multiple components and sub-systems. Engine applications, vibration data use and Artificial Neural Networks (ANNs) used for regression were applied consistently over time. The accuracy of ML algorithms were largely unreported, although some studies reported high accuracy ANN models on training set sizes ranging from 2400 to 31000 samples (Basurko & Uriondo, 2015; Farag, 2017). One of the studies compared multiple ML techniques (Cipollini et al., 2018) remarking that ANNs were a worthwhile approach. Obtaining data was a challenge reported by Cipollini et al. (2018) and prevented Gkerekos et al. (2016) from applying their methodology to a vessel. It was mentioned previously that ML algorithms may be used to distinguish multiple faults, although less than half of the studies created classification-based methods and that only Basurko and Uriondo (2015) developed a method which involves multi-class classification.

A multi-class ML algorithm is adopted in the present study. Regression requires more data than that which could be obtained for the development of the present maintenance system.

Table 2-1: Highlighted Studies - CM and ML Details

			CM Dataset						**ML Algorithm**				
Reference	Application Level	Application	Vibration	Strain	Temperature	Pressure	Other	Quantity	ANN	SVM	RLS	KNN	BGP
Sebastiani et al. (2013)	Component (Multiple, Separate)	6 axial exhaust fans, 2 main cooling pumps, 1 belt-driven exhaust fan, 1 starting air compressor, 2 bearings of propulsion motor, 6 thrusters, 1 shaft bushing	X		X	X	X	Not specified	None				
Cardona-Morales et al. (2014)	Component (Multiple, Separate)	Ship driveline (Engine, gearbox, shaft)	X					Not specified	None				
Basurko and Uriondo (2015)	Component	Medium-speed diesel engine					X	>31300 samples, 45 measurements	X				
Soliman et al. (2015)	Component	Aluminium Hull of catamaran		X				Not specified	None				
Coraddu et al. (2016)	Component	Data Driven Model of Naval Propulsion System (Gas Turbine (GT), GT compressor, hull and propeller					X	> 200 samples, 16 measurements		X	X		
Gkerekos et al. (2016)	Component	Wind turbine pinion gear	X					Not relevant		X			
Farag (2017)	Environment	Environment around vessel					X	> 2400 samples	X				
Lazakis et al. (2018b)	Component (Multiple, Separate)	Panamax container ship main engine	X	X	X		X	Not specified, 12 sensors	X				
Cipollini et al. (2018)	Sub-system	Data Driven Model of Naval Propulsion System (Gas Turbine (GT), GT compressor, hull and propeller			X	X	X	> 400 samples, 25 measurements	X	X	X	X	X
Lazakis et al. (2018a)	Sub-system	Diesel Generator set and supporting components			X	X	X	771 measurements		X			

Table 2-2: Highlighted Studies - Objective and Method

Reference	Objective	Binary Classification	Multi-class classification	Regression
Sebastiani et al. (2013)	Detect anomalous operation of machinery	X		
Cardona-Morales et al. (2014)	Identify signatures which indicate faults	X		
Basurko and Uriondo (2015)	Detect one of four faults in the turbine		X	
Soliman et al. (2015)	Predict fatigue damage and fatigue life using fatigue reliability			X
Coraddu et al. (2016)	Model and predict performance decay of numerical model			X
Gkerekos et al. (2016)	Detect anomalous operation of machinery	X		
Farag (2017)	Predict propulsion power to optimise energy efficiency			X
Lazakis et al. (2018b)	Predict Main Engine Exhaust Gas Temperature			X
Cipollini et al. (2018)	Predict condition of equipment, evaluate performance of different ML methods			X
Lazakis et al. (2018a)	Detect anomalous operation of machinery	X		

Numerous factors preventing the development of maintenance practise were discussed previously in Chapter 1. Discussion of existing studies in the present Chapter 2 highlighted additional factors which inhibit the development of a system using CM and ML techniques. These include: identifying measurements which describe faults, initial and ongoing costs of data collection, data collection in a marine environment, interpreting results of classification, ongoing personnel training and application management costs. Each of these additional factors are discussed in the following sub-sections.

2.1.2. Designing a Data Collection Regime

Each application requires an investigation to determine which ML algorithm should be applied. This investigation also determines the quantity and type of data which must be collected as enough high-quality data must be obtained for use with the chosen algorithm. A large variation on the quantity of data used for the development of ML algorithms was shown previously in Table 2-1. Identifying a suitable ML algorithm and designing experiments to collect data requires specialist expertise, time and additional upfront investment as the necessary expertise is not

widespread within organisations which maintain marine vessels. External contractor involvement may be necessary in addition to specialist equipment.

The present system development was not affected by data collection challenges as the required expertise and resources were available. It is expected that data acquisition shall generally remain a challenge, especially in industry applications, unless provisions are made to acquire the necessary expertise through training personnel or hiring contractors.

2.1.3. Application Cost

The quantity and quality of data required for an acceptable level of classification performance directly influences the cost of developing and managing a CM and ML maintenance system.

CM equipment is expensive. To manage expense for a CM application, it is necessary to select measurement techniques which provide a wealth of descriptive data using as few measurements as possible. Table 2-1 showed that vibration analysis has been applied consistently to date in marine vessel applications as a solution. Vibration analysis is a common technique used in many fields, particularly for the fault diagnosis of rotating equipment (Lee et al., 2014). It is a sensitive technique which provides a wealth of information as displacement, velocity or acceleration waveform measurements. These can be analysed in the time domain or transformed into the frequency domain by applying the Fast Fourier Transform (FFT) method to yield peaks at characteristic frequencies. One vibration waveform can be transformed using the FFT method and analysed to determine equipment condition without other supporting measurements. Accordingly, vibration waveforms are obtained for the shipboard pump application in Chapter 3 and are analysed using a classification algorithm in Chapter 4.

Large datasets are expensive. The cost and time required to obtain a suitable dataset limits the scope of a CM application, especially in the case of multi-class classification algorithms. The size of the dataset can be estimated based on the application. It may be desirable to manage expense by limiting the dataset size for a given application. However, the performance of the ML algorithm may be reduced by limiting dataset size if the algorithm requires more data. The

minimum dataset size is a characteristic of each algorithm. The Naive Bayes classification algorithm generally performs well irrespective of dataset size (Li & Jain, 1998; Zhang, 2005), while Artificial Neural Networks (ANNs) and Support Vector Machines (SVM) may not as the implementation of these requires the development of numerical rather than probabilistic functions. Naive Bayes may seem an obvious choice, though its implementation requires that the assumption of feature independence. The potential effects of this assumption on Naive Bayes performance remain disputed (Stephens et al., 2015), although it is obvious that many features describing the same physical system will be dependent.

Statistical methods also show excellent performance given limited data as models are built based on statistical parameters of the available dataset. Multivariate Analysis (MVA) is a statistical method which allows for removal of redundant information, classification and visualisation of correlated data in any number of dimensions (Duda et al., 2012; Fukunaga, 1990). The only assumption required is that the data come from a normal distribution. In the present work, MVA is applied to create four discriminant classifiers for the pump application in Chapter 4. MVA is described in Section 2.1.3.2, while further detail may be found references such as Duda et al. (2012) and Fukunaga (1990).

When there is no obvious choice of algorithm, multiple algorithms should be compared using performance estimation tools. Existing tools are discussed in the following section.

2.1.3.1. Performance Estimation of Supervised Machine Learning Algorithms

Performance estimation tools are used to compare algorithms and combinations of parameters to ultimately select the algorithm and parameters which provide the best possible performance. It is ideal and efficient to calculate one value for each algorithm or combination of parameters to estimate performance. Unfortunately, researchers in this field have not yet derived this ideal performance estimation tool.

A recent review (Ballabio et al., 2018) discussed existing performance estimation methods and their applicability toward supervised classification algorithm selection and calibration.

Supervised algorithms are trained using learning data with known group names while unsupervised algorithms are trained using data with no known group names. The authors initially discussed group or class-related measurements derived from the confusion matrix. These were sensitivity, specificity and precision and derivatives of these measures. They then discussed 17 global performance estimators derived from the secondary estimators.

Global performance measures included accuracy; defined as the percentage of correct classifications, Cohen's Kappa coefficient (Cohen, 1960), the Matthews Correlation Coefficient (Matthews, 1975), and lastly the area under the Receiver Operating Curve (ROC). ROC is a plotted measure of true and false positive rates used frequently in the case of binary classification problems.

The authors defined benchmark values for each measure, which were based upon random classification using linear discriminant analysis and a combination of 31 different datasets and three scenarios. The three scenarios were classification of all 31 datasets considering: equal apriori probabilities of the classes, class-weighted apriori probabilities and lastly that all samples are assigned to the class with the largest number of samples.

The results of these tests indicated that the value of the global performance estimators changed based on the dataset, and between binary and multi-class problems. They concluded that '… to evaluate the performance of a single classification model, a critical analysis of the confusion matrix should always be performed, in particular when misclassifications have not the same cost'. Thus, a wide variety of performance estimation approaches have been developed to date though none are universally applicable.

It is common practice to use a confusion matrix to determine the number of correct classifications in each class (Comfort et al., 2018; Gholami et al., 2018; Gupta et al., 2018; Turgut et al., 2018). Additional measures such as precision and recall are commonly derived from this matrix in the case of two classes only. These were evaluated in the previous review (Ting, 2017).

A confusion matrix is a square matrix with its dimension representing the number of classes in the problem. The value in element (i, j) of a confusion matrix represents the percentage of samples with known class i which have been classified as j. Therefore, correct classifications are shown on the main diagonal of the confusion matrix as $i = j$, and incorrect classifications are displayed in off-diagonal elements where $i \neq j$.

While a confusion matrix can be computed for any supervised machine learning algorithm, it is not straightforward to compare the performance of different algorithms using this approach as the reasons behind the classification or mis-classification are not always clear. It is also not optimal to use this approach as various algorithms and parameter combinations must be efficiently and consistently evaluated.

Most crucially, information about the most probable fault is the only information used in both performance estimation and further calculations. In some scenarios, information about only the most probable fault is required, while information about all faults must be considered in others, such as the present pump RBM system. Two examples are discussed in the following subsection.

I. The Purpose of Classification

The development of supervised classification algorithms as part of the overall field of ML was intended to reduce the human workload required to create an accurate program for a particular situation (Carbonell et al., 1983). A simple description of such an algorithm is that it transforms some input vector of relevant data into the output, which is a label indicating the most probable class.

The class label output has become the status quo. It is counted and displayed as a confusion matrix and analysed according to various means to estimate the performance supervised classification algorithms as discussed previously. The purpose and intention of the class label output is to define which class the input measurement belongs to. Its value varies when considering the following examples: image recognition for an autonomous vehicle and fault

detection of machinery. The latter is discussed as it relates to the RBM system developed within the present Thesis.

I.I. Image Recognition for Autonomous Vehicles

The purpose of a self-driving car or autonomous vehicle is to transport people and/or goods between two points safely and efficiently, without the need for human intervention. Issues surrounding autonomous vehicles are discussed in greater detail in Brenner and Herrmann (2018). Autonomous vehicle technology involves image recognition to detect objects surrounding the vehicle. Supervised classification algorithms can be used to achieve this, processing images as the vehicle moves and classifying objects in the images as hazards, road markings or otherwise (Huval et al., 2015) by determining the class label. In this case, the class label is valuable as the software behind it is required to efficiently process a constant stream of data and rapidly respond to the outcome by manoeuvring the vehicle.

II.I. Fault Detection of Machinery

The faults developing within a piece of machinery at a given time are unknown prior to human inspection or maintenance is performed, unless CM hardware and software have been installed. This is rare aboard commercial vessels, and an unpublished statistic in terms of Naval vessels.

It was discussed previously how fault detection of machinery may be performed using supervised classification algorithms, noting that few multi-class marine applications exist. The example discussed is a general pump within the marine context as this relates to the application within the present work.

A supervised classification algorithm can be used to produce a class label from some pump measurement vector input to highlight which ‘fault found’ or ‘no fault found’ state is occurring. Then, another module in the maintenance system or an expert must decide upon a suitable course of action.

A single label discounts the value of the other ‘fault found’ or ‘no fault found’ states, which may be highly developed. This additional information must be considered to ensure that the best possible maintenance decisions are made. A maintenance system which considers the most probable class only also loses the ability to monitor and predict the development of faults as the system shall only identify when a fault *has occurred.* Further, consideration all ‘fault found’ and ‘no fault found’ probabilities in subsequent calculations is necessary to determine the best possible maintenance action.

II. A New Approach

It is possible to use a supervised classification algorithm to compute the probabilities of all classes. In fact, this calculation may occur prior to the class label assignment in some algorithms. If not, any supervised algorithm can be modified to produce these probabilities if it is Bayesian, or approximate quantities if it is not. In an RBM system, calculation of all class probabilities represents full quantification of the risk involved in operation and maintenance as part of the Risk Assessment step.

A performance estimation method is required to determine the meaning of these probabilities. The existing confusion and certainty matrices are discussed in this regard in the following sections alongside a new tool which demonstrates how probabilities can be transformed to estimate the doubt of a Bayesian algorithm. Approximate certainty and doubt matrices are also presented for non-Bayesian algorithms.

III. Understanding Supervised Classification Algorithm Behaviour

I.III. Supervised Bayesian Classification Algorithms

To develop interpretation tools, Bayesian classifiers should be considered initially as they can produce probabilities. A supervised Bayesian Classifier generally follows the procedure below:

Given a measurement vector $\vec{x}$ which belongs to class k determine:

1. The set of posterior probabilities $P(c_k \mid \vec{x})$ for all classes c_k

2. The most probable class k_{Opt} where: $k_{Opt} = \arg\max P(c_k \mid \vec{x})$

3. Assign k_{Opt} as the class label.

To date, classification stops after the assignment of k_{Opt}, and the classification is complete.

A supervised Bayesian classifier can also be used to calculate the following additional class rankings:

$$k_l = \arg\max P(c_k \mid \vec{x}) \tag{2-1}$$

In (2-1), k_l is the rank of each of the remaining classes where $k_l \neq k_{Opt}$ and the rank number suffix $l = 2,3...k$. k_{Opt} shall always be assigned rank 1 with $k_1 = k_{Opt}$ as it is the optimal class. For a hypothetical 3 class example where $k = 3$, consider that $k_{Opt} = c_3$ is obtained. Then, $k_1 = k_{Opt} = c_3$ and it may be determined using (2-1) that the second and third class ranks are either $k_2 = c_1$ and $k_3 = c_2$ or alternatively $k_2 = c_2$ and $k_3 = c_1$.

The remaining class probabilities $P(c_k \mid \vec{x})$ and their ranks k_l show how the classifier behaves toward each class specifically and can be used to derive new performance estimation tools.

II.III. <u>Supervised Maximum-Likelihood Classification Algorithms</u>

The former Bayesian approach can be adapted for any maximum-likelihood classifier, considering that k_{Opt} and analogous values to $P(c_k \mid \vec{x})$ are determined in a similar way:

Given a measurement vector $\vec{x}$ which belongs to class k, determine:

1. The set of function evaluations $f(\vec{x} \mid c_k)$ for all classes c_k

2. The most probable class k_{Opt} where: $k_{Opt} = \arg\max f(c_k \mid \vec{x})$

3. Assign k_{Opt} as the class label.

In this case, the additional class ranking k_l is described in (2-2) as a maximum-likelihood classifier which maximizes a function value rather than probabilities:

$$k_l = \arg\max f(c_k \mid \vec{x}) \tag{2-2}$$

$k_1 = k_{Opt}$ is true as k_{Opt} is the optimal class. In (2-2), k_l is analogously the rank of each of the remaining classes where $k_{Opt} \neq k_l$ and $l = 2,3...k$. Further, the pseudo-posterior probability $\tilde{P}(c_k \mid \vec{x})$ can be calculated from a maximum-likelihood classifier as the weighted fraction of the function evaluation $f(c_k \mid \vec{x})$ using (2-3):

$$\tilde{P}(c_k \mid \vec{x}) = \frac{f(\vec{x} \mid c_k)}{\sum_{1}^{k} f(\vec{x} \mid c_k)} \tag{2-3}$$

III.III. <u>A General Supervised Classification Algorithm</u>

For the general case of any supervised classifier, a relevant distance measure ε_k for each class can be used to transform measurements into a pseudo-posterior probability $\tilde{P}(c_k \mid \vec{x})$ using the following approximation:

$$f(\vec{x} \mid c_k) \sim \frac{1}{\varepsilon_k} \tag{2-4}$$

The interpretation of ε_k is defined by the classifier. ε_k could be assigned as any one of: the distance of $\vec{x}$ to the class mean μ_k ; the distance to any number of nearby points as used in k-nearest neighbour (KNN) classifiers; or the distance to the class division vector as used in support vector machine (SVM) classifiers. Principles of these classification methods are discussed in Li et al. (2018). ε_k could be measured in any number of ways appropriate to the nature of the classification problem, including Euclidean principles, Mahalanobis distance (Weinberger et al., 2009)

Minowski distance, new variants (Merigo & Casanovas, 2011) or problem-specific measurements such as inner distance (Ling & Jacobs, 2007).

In the case of ANNs, a suitable approximation of $f(\vec{x} \mid c_k)$ in relation to the neural evaluation function $g(c_k)$ is given in (2-5):

$$f(\vec{x} \mid c_k) \sim g(c_k) \tag{2-5}$$

As ε_k can be assigned and $f(\vec{x} \mid c_k)$ approximated, pseudo-probability values can be determined by applying (2-5) in the general case of any supervised learning algorithm.

Consideration of the remaining class probabilities and class pseudo-probabilities led to the development of the performance estimation tools in the following section.

IV. Certainty and Doubt Matrices

The formal definitions of the *certainty matrix* and the *doubt matrix* are presented in this section for supervised Bayesian classification algorithms; alongside the *pseudo-certainty matrix* and *pseudo-doubt matrix* for all other supervised classification algorithms. These performance estimation tools are to be used alongside the confusion matrix to estimate the performance of any supervised classification algorithm.

The *certainty matrix* was postulated by Nedev and Tenekedjiev (1994) for a supervised Bayesian classification algorithm as a square matrix which aims to compute the certainty a supervised algorithm has in assigning a sample to a given class. The matrix dimension is equal to the number of classes, i. The value in element (i, j) of a certainty matrix represents the percentage of certainty that the classifier has in assigning samples with known class i to class j. The percentage certainty for element (i, j) is calculated as the mean probability that samples with known class i are assigned to class j. 'Not a number' or NaN is returned when there are no calculations required for the combination of (i, j).

Certainty is dictated by the learned parameters in the classifier. A classifier should possess high values on the main diagonal as this encourages correct classifications (where $i = j$) and lower values in all other elements to discourage incorrect classifications. In summary, the certainty matrix serves to highlight the confidence the classifier has in its correct classifications.

The *doubt matrix* is defined in the present Thesis for a supervised Bayesian classification algorithm as a square matrix of dimension i which aims to compute the doubt that a supervised learning algorithm has in assigning a sample of class i to class j. The value in element (i, j) of a doubt matrix represents the percentage of doubt that the classifier has in assigning samples with known class i to class j. The percentage doubt for element (i, j) when $i = j$ is calculated as the mean probability that samples with known class i were assigned to class j, where $j \neq i$ and j has the maximum probability of all j. The percentage doubt for element (i, j) when $i \neq j$ is calculated as the mean probability that samples with known class i are assigned to class i, which can be interpreted as the percentage that the classifier doubts the true and assigned classes are equal. 'Not a number' or NaN is returned when there are no calculations required for the combination of (i, j).

Doubt is also dictated by the learned parameters in the classifier. A classifier should possess lower doubt values on the main diagonal as this encourages correct classifications (where $i = j$) and higher values in all other elements to discourage incorrect classifications. In summary, the doubt matrix shows the degree to which the classifier suppresses incorrect classifications.

It cannot be assumed that $certainty(i, j) = 1 - doubt(i, j)$ and other probability rules hold as computations involve different numbers of samples in each combination of (i, j) and their mean values.

Probability values can only be determined using Bayesian approaches. Hence, the certainty and doubt matrices described above relate only to supervised Bayesian classification algorithms. Similar matrices could be produced using pseudo-probability values for all other supervised learning algorithms as discussed in the previous section. Certainty and doubt matrices created for non-Bayesian algorithms are termed the *pseudo-certainty matrix* and the *pseudo-doubt matrix*.

Performance estimation of the pump RBM system classifiers is performed using confusion, certainty and doubt matrices in the following Chapter 4.

2.1.3.2. Multivariate Analysis and Discriminant Function Classification

The application of Multivariate Analysis (MVA) to create a linear discriminant function classifier requires only the assumption that data are multi-normally distributed. This is a reasonable assumption as there are many examples of the normal distribution occurring in nature, such as in the measurement of the heights of 100 random people. It is also straightforward to transform skewed data so that they become multi-normal. Such data may result due to an insufficient sample size.

MVA is applied in the present Thesis to create k discriminant functions $g_i(\vec{x})$ for each class where $i = 2 \ldots k$ and $\vec{x}$ is a vector of measurements describing the condition of a piece of equipment. This process is also known as Linear Discriminant Analysis (LDA). $g_i(\vec{x})$ can be visualised as $n-1$ dimensional objects separating the classes of $\vec{x}$ in n-dimensional space.

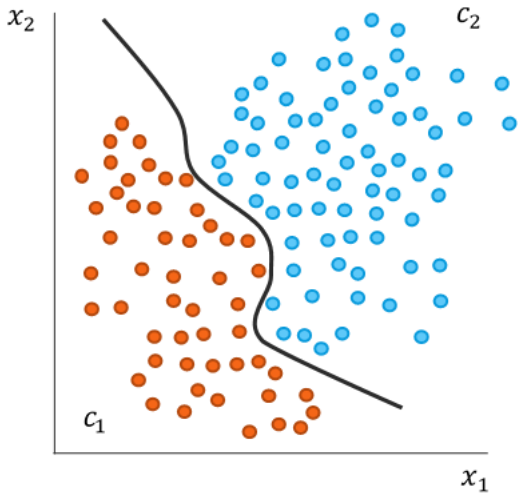


Figure 2-1: Discriminant Function Concept where n = 2

In Figure 2-1, $i = 2$ and $\vec{x} = [x_1, x_2]$. Therefore, data in k classes (c_1 red) and (c_2 blue) line on a plane, while the discriminant functions $g_1(\vec{x})$ and $g_2(\vec{x})$ are overlapping lines separating these data points. The set of k functions $g_1(\vec{x}),\ldots g_k(\vec{x})$ forms the classifier. The classifier will assign vector $\vec{x}$ to class k if evaluation of $g_k(\vec{x})$ produces the maximum value of all $g_1(\vec{x}),\ldots g_k(\vec{x})$. Considering each class separately, it is assumed that the data represented by $\vec{x}$ are multi-normally distributed and can be described by the general multivariate normal density (Duda et al., 2012) given in (2-6).

$$p(\vec{x}) = \frac{1}{(2\pi)^{d/2}|\Sigma|^{1/2}} e^{\left(-\frac{1}{2}(\vec{x}-\vec{\mu})^t \Sigma^{-1}(x-\vec{\mu})\right)} \tag{2-6}$$

In (2-6), $\vec{\mu}$ is the vector of mean feature values μ_f in n samples of $\vec{x}$ calculated for each feature f by applying (2-7).

$$\mu_f = \frac{1}{n}\sum_{i=1}^{n} x_f \tag{2-7}$$

Σ is the variance-covariance matrix of all n samples of $\vec{x}$. Σ directly incorporates the correlation of the features in $\vec{x}$ into the multivariate density as the product of their respective variances or

covariances. In the case of two features x_1 and x_2 illustrated previously, initially $\vec{\mu} = [\mu_1, \mu_2]$ must be calculated using (2-7), to determine the variance-covariance matrix Σ as follows:

$$\Sigma = \begin{bmatrix} \sigma_1^2 & r_{1,2}\sigma_1\sigma_2 \\ r_{2,1}\sigma_1\sigma_2 & \sigma_2^2 \end{bmatrix}$$

Where σ_f is the standard deviation or square root of the variance of sample x_f^n given by (2-8)

$$\sigma_f^2 = \frac{1}{n-1} \sum_{i=1}^{n} \left(x_f^n - \mu_f \right) \tag{2-8}$$

And the correlation coefficients represented by $r_{1,2}$ and $r_{2,1}$ lie within the range of $-1 < r_{l,m} < 1$ and are given by (2-9):

$$r_{l,m} = \frac{\frac{1}{n-1} \sum_{i=1}^{n} \left(\left(x_l^n - \mu_l \right) \left(x_m^n - \mu_m \right) \right)}{\sigma_l \sigma_m} \qquad \text{and} \qquad l \neq m \tag{2-9}$$

To determine the posterior probability $P(c_k \mid \vec{x})$ considering all classes c_k, Bayes Theorem for Continuous Variables in (2-10) can be incorporated to develop the discriminant functions $g_i(\vec{x})$.

$$P(c_k \mid \vec{x}) = \frac{p(\vec{x} \mid c_k) \; P(c_k)}{\sum_{i=1}^{k} p(\vec{x} \mid c_k) \; P(c_k)} \tag{2-10}$$

Note the evidence term in the denominator of (2-10) is the weighted sum of the class membership of $\vec{x}$ in the case of discrete variables. This can be removed as it is essentially a scaling constant and the logarithm of the expression evaluated to develop the general linear form of $g_i(\vec{x})$ in (2-11).

$$g_k(\vec{x}) = \ln p(\vec{x} \mid c_k) + \ln P(c_k) \tag{2-11}$$

The natural logarithm of (2-6) is then substituted into (2-11) to give a general expression for $g_k(\vec{x})$ considering the multivariate distribution shown in (2-12).

$$g_k(\vec{x}) = -\frac{1}{2}(\vec{x} - \vec{\mu}_k)^t \Sigma_k^{-1}(x - \mu_k) - \frac{d}{2}\ln 2\pi - \frac{1}{2}\ln|\Sigma_k| + \ln P(c_k) \quad (2\text{-}12)$$

Assuming the *k* covariance matrices are identical but arbitrary i.e. give the classes the same shape, the result is the linear discriminant function in (2-13).

$$g_k(\vec{x}) = -\frac{1}{2}(\vec{x} - \vec{\mu}_k)^t \Sigma_k^{-1}(x - \mu_k) + \ln P(c_k) \quad (2\text{-}13)$$

Further, (2-13) can also be expressed as (2-14):

$$g_k(\vec{x}) = \boldsymbol{w}_k^t \vec{x} + \boldsymbol{w}_{k0} \quad (2\text{-}14)$$

Where: $\boldsymbol{w}_k = \Sigma^{-1} \vec{\mu}_k$ and $\boldsymbol{w}_{k0} = -\frac{1}{2} \vec{\mu}_k^{\,t} \Sigma^{-1} \vec{\mu}_k + \ln P(c_k)$.

$\boldsymbol{w}_k$ and $\boldsymbol{w}_{k0}$ are the learned parameters from the dataset using this classification method and where the overall covariance matrix Σ is the combination of individual class covariance matrices Σ_k according to (2-15).

$$\Sigma = \sum_{i=1}^{k} \frac{(n_k - k)\sum_{i=1}^{k}\left(\sum \boldsymbol{WC}_k\right)\Sigma_k}{\sum n_k} \quad (2\text{-}15)$$

In (2-15), k is the number of classes, n_k is the number of samples in class k and incorporate weighting coefficients $\boldsymbol{WC}_k$. Weighting coefficients allow the incorporation of similar data from other classes when there is insufficient data in class k . $\boldsymbol{WC}_k$ is defined as the fuzzy consistency coefficient in (Tenekedjiev et al., 2006).

The posterior probabilities of each class $P(c_k | \vec{x})$ may be calculated using (2-16):

$$P(c_k | \vec{x}) = \frac{1}{\sum_{i=1}^{k} e^{(g_i(\vec{x}) - g_k(\vec{x}))}} \quad (2\text{-}16)$$

This calculation effectively measures the exponential distance between the point $g_i(\vec{x})$ and $g_k(\vec{x})$ to determine membership of that class, and is less computationally intensive than an equivalent representation presented in (2-17). The derivation of (2-17) is shown in Appendix A.

$$P(c_k \mid \vec{x}) = \frac{e^{(g_k(\vec{x}))}}{\sum_{i=1}^{k} e^{(g_k(\vec{x}))}} \tag{2-17}$$

Lastly, the probabilities resulting from the linear discriminant classification approach can be interpreted using confusion, certainty and doubt matrices.

2.1.3.3. Whitening Transformation

Σ^{-1} must be computed and therefore Σ must not be degenerate. If Σ is degenerate, this implies that correlations between the coefficients, measured by $r_{l,m}$ must be reduced. In practice, redundant features must be removed and new uncorrelated features must be created. This can be achieved using a whitening transformation (Duda et al., 2012), applying the transformation matrix in (2-18) to the learning data.

$$T = \begin{bmatrix} \vec{V_1} / \lambda_1 \\ \vdots \\ \vec{V_n} / \lambda_n \end{bmatrix} \tag{2-18}$$

Eigenvalues λ_i and eigenvectors $\vec{V_i}$ are derived from Σ. Application of this transformation to Σ and $\vec{\mu}$ gives a unit hypersphere in a new co-ordinate system which is called the objective space, parameterised by Σ_{obj} and $\vec{\mu}_{obj}$. Calculations of these new parameters are achieved using the following equations:

$$\vec{\mu}_{obj} = T(\vec{\mu}_k - \mu) \tag{2-19}$$

$$\Sigma_{obj} = T \Sigma T' \tag{2-20}$$

In the computation of T, some eigenvalues are may be zero, which means that the number of features within the objective space can be reduced while retaining their information. This transformation also allows visualisation in the objective space using no more than three eigenvalues at a time, or complete visualisation of the data if Σ is $n-3$ times degenerate in the objective space from the original n dimensional feature space.

Due to the whitening transformation, all classifications of $\vec{x}$ are performed in the objective space.

2.1.3.4. Interpretation of Results

While CM and ML classification can be used to determine the faults within a system using a multi-class classification approach, these techniques produce probabilities as opposed to actionable outcomes. As mentioned previously, specialist expertise is required to interpret these probabilities into outcomes using a decision-making process. This interpretation may be achieved by the specialist using their experience, although the specialist can only make decisions based on the most probable fault. This means that some information is lost, and has other disadvantages as discussed in Chapter 1. All information must be considered in the Maintenance Scheduling element of an RBM system. Decision Theory was recommended for this purpose in Chapter 1. Current decision-making approaches and Decision Theory are discussed in the second part of this Chapter, beginning with the following Section 2.2.

2.2. Modelling Decisions for Marine Vessel Maintenance

Methods which gather and analyse data to automatically identify potential faults within equipment were discussed in Section 2.1.1. This discussion highlighted that the identification of maintenance actions and their optimal timing remains an inefficient and subjective process. There is a need for automated decision-making in the maintenance context. Automated decision-making will enable the identification of the optimal time to perform maintenance and ensure the consistent selection of the best possible maintenance action. The combination of CM, ML and an

automated decision-making method will result in a comprehensive system which can analyse equipment to automatically produce actionable results for maintainers.

2.2.1. Existing Applications of Decision-making Techniques to Marine Vessel Maintenance: *Trends and Challenges*

Algorithms and methods for automated interpretation of data model the human decision-making process, which often involves multiple criteria. Decisions considering multiple criteria are often modelled using Multi-Criteria Decision-making methods (MCDM). MCDM in the maintenance context remains a developing field (de Almeida et al., 2015). 'Decision-making' and MCDM (de Almeida et al., 2015) are generally used to select a maintenance framework such as periodic PM, RCM or CBM (Ahmadi et al., 2010; Ghosh & Roy, 2010; Pariazar et al., 2008; Sarkar & Behera, 2012; Shahin et al., 2012), but not to identify maintenance actions. Alternatively, Fuzzy Logic can be used to describe decisions using continuous or overlapping groups rather than distinctive ones (Klir & Yuan, 1995). Marine logistics applications have been discussed, (Ries et al., 2017) although these excluded maintenance decisions. A review of the literature has revealed only a small number of studies applied MCDM or other techniques to the maintenance of marine vessels. These studies are tabulated in Table 2-3.

Table 2-3: Highlighted Studies - Decision Support and Decision-making

Reference	Objective	Application	Methodology	Single Result	Multiple Results	Criterion	Single DM	Multiple DMs
Lagoudis et al. (2006)	Determine the best performing business sector of the ocean transport industry (liner, dry bulk, liquid bulk and specialised)	Economic Environment	MAUT	X		Quality, Service, Cost, Cycle Time		X
Nguyen (2008)	Select the best material to construct a future vessel, considering steel, light-weight aluminium, fiberglass composite and carbon-fibre composite	Vessel Design	MAUT	X		Groups of attributes describing performance, operation and cost		X
Goossens and Basten (2015)	Determine the optimal maintenance Policy for a vessel, considering failure-based, time-based and condition-based maintenance.	Vessel	Analytical Hierarchy Process (AHP)	X		Clusters of criteria		X
Emovon (2016)	Determine the optimal maintenance Policy for a vessel, considering corrective maintenance, scheduled overhaul, scheduled replacement, continuous on-condition tasks and scheduled on-condition tasks for a seawater pump as part of the central cooling system of a marine diesel engine.	Component	Combined method - Delphi, AHP and Technique for Order of Preference by Similarity to Ideal Solution (TOPSIS)	X		Twelve criteria		X
Nguyen (2017)	Find optimal start time of maintenance cycle for each ship in one of two squadrons to ensure 75% fleet availability in maintenance cycle. Constraints are minimum availability is 50% and one dockyard are used.	Multiple fleets of single vessel class	Multi-objective Optimisation - Differential Evolution (DE) Algorithm		X	Ship Availability	X	
Sabatino (2017)	Determine optimal structural health monitoring (SHM) duration considering uniform or non-uniform monitoring intervals, within cost and duration constraints.	Vessel	MAUT and Multi-Objective Optimisation - Genetic Algorithm (GA)		X	Condition data, cost to obtain data, availability, number of monitoring cycles, shape factor of utility function	X	
Williams and Hester (2017)	Compare vessel maintenance plans to determine if maintenance deferrals are possible for one of two vessels.	Fleet (Two vessels)	MAUT	X		Utility values for different types of maintenance: mandatory maintenance, non-mandatory maintenance, mission impact from maintenance, mission impact from planned modernizations, and maintenance backlog.	Not specified	

Earlier studies focused on obtaining a single result using Multi-Attribute Utility Theory (MAUT) and the Analytical Hierarchy Process (AHP) (Saaty, 2012) across a variety of marine problems. Recent studies continued the trend of applying MAUT (Williams & Hester, 2017) or alternatively focused on generating multiple solutions using optimisation algorithms (Nguyen, 2017; Sabatino, 2017) for maintenance problems. There is a trend toward decisions made using large amounts of criteria and multiple decision-makers (DMs), which highlights the complexity of these problems. These trends have highlighted that numerous factors require consideration, these include: the selection of an appropriate technique, the number of DMs, the expertise of the DMs, and the nature of the result to achieve the goal of automated data interpretation. These factors are discussed in the following sub-sections.

2.2.2. Techniques

Multiple outcomes resulting from optimisation and complex attribute weight derivations required for Analytical Hierarchy Process (AHP) applications increase the amount of resources required. A simple and efficient methodology is recommended to minimise overhead costs in the marine context. It is also recommended that MAUT is incorporated within future maintenance systems as it allows for the direct quantification of preferences, though it must be adapted to consider uncertainty in the real case. MAUT is discussed in the following sub-section.

2.2.2.1. Applications of MAUT

The application of MAUT has been common among these studies. MAUT quantitatively models a decision-maker's (DMs) preference structure across multiple attributes or criterion, and results in a ranked list of alternatives(Clemen, 1996; French, 1986; Keeney & Raiffa, 1993). MAUT was not applied to marine vessels until recently (de Almeida et al., 2015).

MAUT is applied in the relevant studies in Table 2-3 as a standalone MCDM approach as it is claimed that MAUT can be applied on its own to model uncertainty in maintenance problems (de Almeida et al., 2015) and vessel maintenance problems specifically (Emovon et al., 2016;

Williams & Hester, 2017). While it is true that MAUT may model preferences under uncertainty, both standalone MAUT application and previously applied methodologies consider that the overall decision is made under certainty, i.e. prior knowledge that a single state of nature is occurring. However, many decisions require that a real DM considers probabilities of different states of nature occurring in addition to how these affect the outcomes of their decisions. These are decisions under uncertainty or decisions under risk. The decision-making process must consider uncertainty and risk in decision-making, especially in the development of the present RBM system.

Decision Theory (Clemen, 1996; French, 1986) describes how a rational DM can combine probability and multi-attribute utility into expected utility using a decision analysis to make his decision under risk. Decision Theory is adopted in the present work, in line with recommendations made in Chapter 1. Decision Theory is discussed further in Section 2.2.5.

2.2.3. Number of Experts and their Expertise

The studies in Table 2-3 tended to involve multiple DMs. It is self-evident that the most effective decision is taken by the DM with the most relevant experience. Studies focusing on decision-making in maintenance should only involve one DM with the most experience as the maintainer of the equipment of study. Other perspectives will not improve the outcome for equipment health and performance.

2.2.4. Selecting a Maintenance Action

None of the studies in Table 2-3 selected maintenance actions to rectify faults in the equipment. A prominent technique was described previously in Section 2.2.2.1, and it was recommended that Decision Theory may be applied as an improvement upon the existing approach. Application of Decision Theory to select a single maintenance action can potentially deliver consistency and efficiency in decision-making beyond periodic PM and RCM. No similar applications exist to date. Decision Theory is discussed further in Section 2.2.5.

2.2.5. Decision Theory

A method to select a maintenance action automatically is required to develop the RBM system within the present Thesis. The selection of a maintenance action is a decision involving uncertainty and risk as each action has multiple possible outcomes. These include repairing the equipment or scheduling maintenance for a later time. This section presents the key elements of Decision Theory which are utilized within the present Thesis, and a novel extension of this theory developed during the present work which can be used to model maintenance deferral actions.

2.2.5.1. Decision-making and Decision Analysis Methods

A DM is required to choose one of multiple courses of action to resolve his decision problem, and he must resolve the decision within externally imposed conditions. Conditions include resource and time constraints as well as uncertainty and ambiguity. DMs can be guided toward rational decisions by following existing approaches (Pratt et al., 2008), and with additional knowledge regarding how to manage their own internal biases and psychological traits (Ariely, 2008).

Decisions considering only one possible situation can be analysed using popular approaches such as the AHP (Saaty, 2012; Saaty & Vargas, 2012), ELECTRE (Govindan & Jepsen, 2016), TOPSIS (Yıldızbaşı & Daneshvar Rouyendegh, 2018) and MCDM methods (Triantaphyllou, 2013). This condition is known as ‘certainty’ as all other possible situations have been disregarded.

In contrast, real decisions are made considering that there are multiple possible outcomes which can occur. This condition is known as ‘uncertainty’. Within the condition of strict uncertainty, decisions can be made according to various criteria such as Laplace, Wald or maximax (French, 1986; Tenekedjiev & Nikolova, 2008), although these criterion may lead to irrational solutions as they do not obey the minimal rationality requirements of choice (Rapoport, 1989). Rationality involves making choices which are consistent with improving one’s situation in some way.

The risk involved in a decision made under the condition of strict uncertainty using probability distributions can be fully quantified using probability distributions.

Von Neumann and Morgenstern (1947) have defined utility theory as part of quantitative Decision Theory (French & Insua, 2000) as one of the most rational approaches to model decisions under strict uncertainty, which are also described as decisions under risk. Analyses of decisions under risk can be completed using one or more decision trees. Uncertain points within the tree as well as outcomes are modelled as lotteries. Further, utility theory prescribes how uncertain alternatives may be ranked when they are modelled as lotteries.

I. Lotteries, Utility and Expected Utility

A lottery l_k consists of prizes or outcomes or consequences $\vec{y}_i^{(k)}$ and their corresponding probabilities of occurrence $p_i^{(k)}$, where $\sum_{i=1}^{t} p_i^{(k)} = 1$. The lottery outcomes or consequences provide a full description of the results of a certain state of nature occurring, where states of nature are represented by $p_i^{(k)}$. The description includes all attributes of each consequence that are important to the DM. A consequence with many attributes is usually described as a multi-dimensional vector whose elements are numerical estimates of the magnitude of each attribute. To solve the lottery, the DM must value each outcome, which is quantified as its utility. Multi-dimensional consequences may be valued in their entirety by the DM or valued by attribute. All attributes are combined according to their scaling constant into an overall multi-attribute utility value. Multi-attribute utility Theory (MAUT) is described in detail in Section 2.2.5.2. The solution of the lottery is calculated as its expected utility, which is determined from a combination of the utility values and probabilities $p_i^{(k)}$.

Lotteries are often described as some chance mechanism that may generate a prize, such as a raffle or game of roulette. A set of lotteries is designated as *L*. At least one uncertain alternative must be present in *L*. *Direct outcomes* occur after one iteration of the chance mechanism, while *ultimate outcomes* occur after multiple iterations of the chance mechanism have resolved all uncertainty – such that all lotteries have been 'drawn'. The final or *ultimate outcomes* of lotteries

within *L* are designated as *Y*. The DM needs to rank each alternative modelled as a lottery within the overall set *L*, by determining the expected utility solution of each lottery.

There are numerous types of lotteries considering the nature of both *Y* and *L*. If these are both countable, outcomes are modelled as Ordinary Lotteries (OL), otherwise outcomes are modelled as one of many other types (Nikolova et al., 2011; Pratt et al., 2008). The present Thesis discusses only OLs.

Further, there are different types of OLs considering their direct outcomes. Lotteries which directly resolve into ultimate outcomes after one iteration of their chance mechanism are termed *simple OLs* as their direct and ultimate outcomes are equivalent.

An example of a simple OL is presented in Figure 2-2, where l_3 is the simple OL, $\vec{y}_1, \vec{y}_2, \vec{y}_3, y_4$ are the prizes it generates with probabilities respectively $p_1^{(3)}, p_2^{(3)}, p_3^{(3)}, p_4^{(3)}$.

Then $l_2 =< p_1^{(3)}, \vec{y}_1;\ p_2^{(3)}, \vec{y}_2;\ p_3^{(3)}, \vec{y}_3;\ p_4^{(3)}, y_4 >$. The signs "<" and ">" indicate the beginning and the end of a lottery.

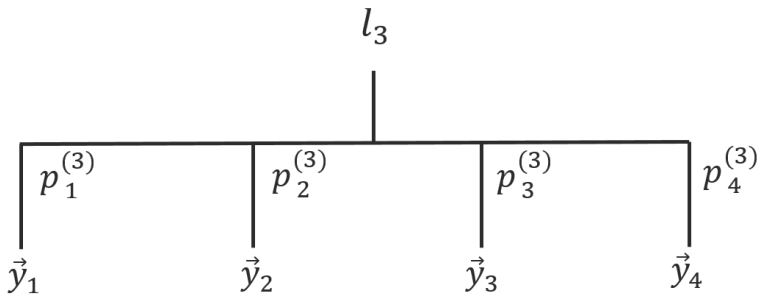


Figure 2-2: Example of a Simple OL

On the other hand, lotteries which do not directly resolve into their ultimate outcomes after one iteration of the chance mechanism are termed *compound OLs*. They may have any number of direct outcome sets prior to the occurrence of their set of ultimate outcomes *Y*. An example of a compound OL is presented in Figure 2-3, where:

$L_1 =< q_1^{(1)}, \vec{y}_1;\ q_2^{(1)}, l_1;\ q_3^{(1)}, \vec{y}_2;\ q_4^{(1)}, y_4;\ q_5^{(1)}, L_2 >$ is the compound OL, $p_j^{(k)}$ denotes the probability of the *j*-th consequence in the *k*-th simple OL, while $q_j^{(k)}$ denotes the probability of the *j*-th element of the direct outcome in the *k*-th compound OL.

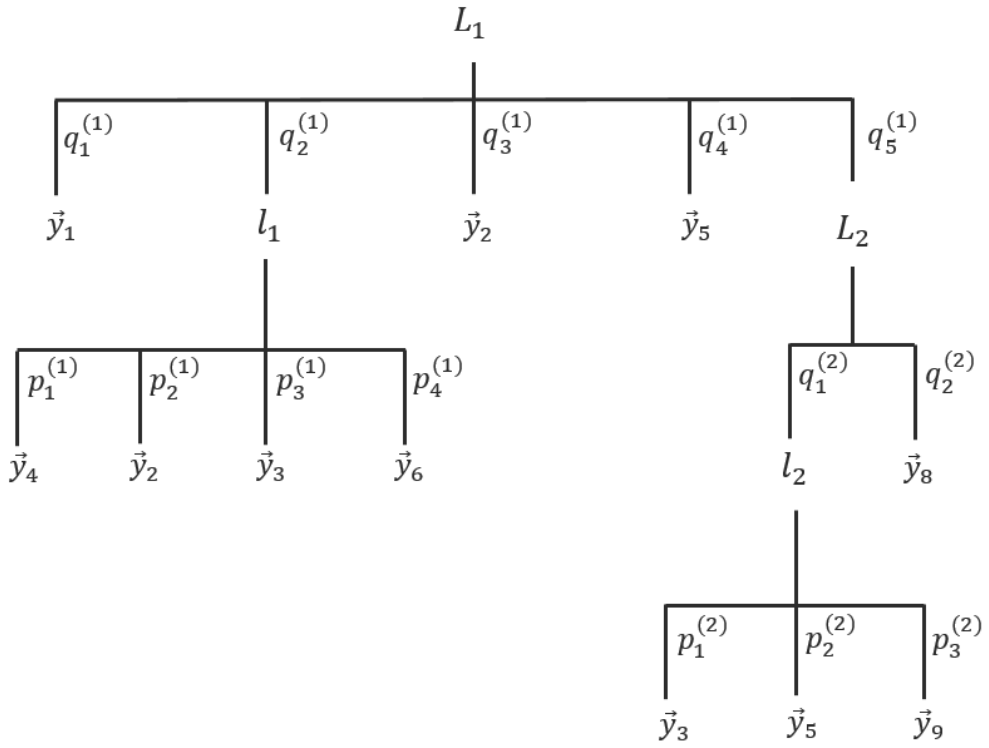


Figure 2-3: Example of a Compound OL

Where the number of chance mechanisms required to resolve a compound OL are less than infinity, the lottery is a Finitely Compound Ordinary Lottery (FCOL), otherwise it is an Infinitely Compound Ordinary Lottery (ICOL). Current theory provides an algorithm for reduction of any FCOL to an equivalent simple OL (French, 1986) whose expected utility solution can be readily determined. However, was not previously the case for ICOLs.

I. Maximum Reduced ICOLs

ICOLs may be used as models for any time-based decision, or as models of unlikely events including rolling a die which lands on a corner. Considering current theory, their unique and complex structures meant that their expected utility solutions could not be determined, thus

limiting their applications in decision analyses. This in turn limited the overall applicability of Decision Theory itself to model decision problems.

An example ICOL is the set of lotteries *L*:

$$L = \left\{ l_1, l_2, l_3, \ldots, l_{n_l}, L_1, L_2, L_3, \ldots, L_{n_L}, L_1^\infty, L_2^\infty, L_3^\infty, \quad , L_{n_{L^\infty}}^\infty \right\} \qquad (2\text{-}21)$$

In (2-21), $l_1, l_2, l_3, \ldots, l_{n_l}$ are simple OLs (n_l in count), $L_1, L_2, L_3, \ldots, L_{n_L}$ are FCOLs (n_L in count), and $L_1^\infty, L_2^\infty, L_3^\infty, \ldots, L_{n_{L^\infty}}^\infty$ are ICOLs (n_{L^∞} in count).

Generally, *L* cannot be reduced to a simple OL due to the additional complexity in the form of quantities n_L and n_{L^∞}, which is further exacerbated by the countless variations possible within the simple OL, compound OL and ICOL structures within *L*. However, it is possible to simplify the ICOL as much as possible by applying Algorithm A.

I.I. <u>Algorithm A:</u> Maximum Reduction of ICOLs

1. Cut all branches of the original ICOL so that each branch then ends with either a consequence or an ICOL. This representation would not be unique.
2. Reduce all branches that end with a consequence to a single simple OL, termed a *fictitious simple OL*.

The result of Algorithm A is a *maximum reduced ICOL*. It contains one *fictitious simple OL* and several ICOLs. The fictitious simple OL may sometimes be a degenerate lottery, which is an ultimate outcome. Outcomes within the fictitious simple lottery are normalized by the sum of their combined weighted probabilities within the original ICOL. A given ICOL may have many possible maximum reduced ICOL equivalents at any given level and many given prizes and lotteries depending on the decision problem.

Use of the reduction algorithm is demonstrated using the ICOL in Figure 2-1. Here, L_1^∞ is an ICOL, and in its structure it has other ICOLs denoted as $L_2^\infty, L_3^\infty, \ldots, L_8^\infty$. Since L_1^∞ is an ICOL, it cannot be reduced to a simple OL by applying current theory.

However, a simple OL may be defined that will have as direct outcomes the prizes $\vec{y}_1, \vec{y}_2, \vec{y}_3, \vec{y}_4, y_5$ in L_1^∞ with their aggregated probabilities to be received from *a given part of* L_1^∞. The resulting fictitious lottery is denoted as l_1^f. One possible structure of L_1^∞ with the l_1^f is given by L_{10}^∞ in Figure 2-5.

The probabilities in l_1^f can be calculated from L_1^∞ using probability theory and the axioms of utility. The aggregated probability to receive each prize from the whole L_1^∞ is calculated as follows: $p(\vec{y}_1) = 0.34$, $p(\vec{y}_2) = 0.056$, $p(\vec{y}_3) = 0.076$, $p(\vec{y}_4) = 0.003$, $p(\vec{y}_5) = 0.416$.

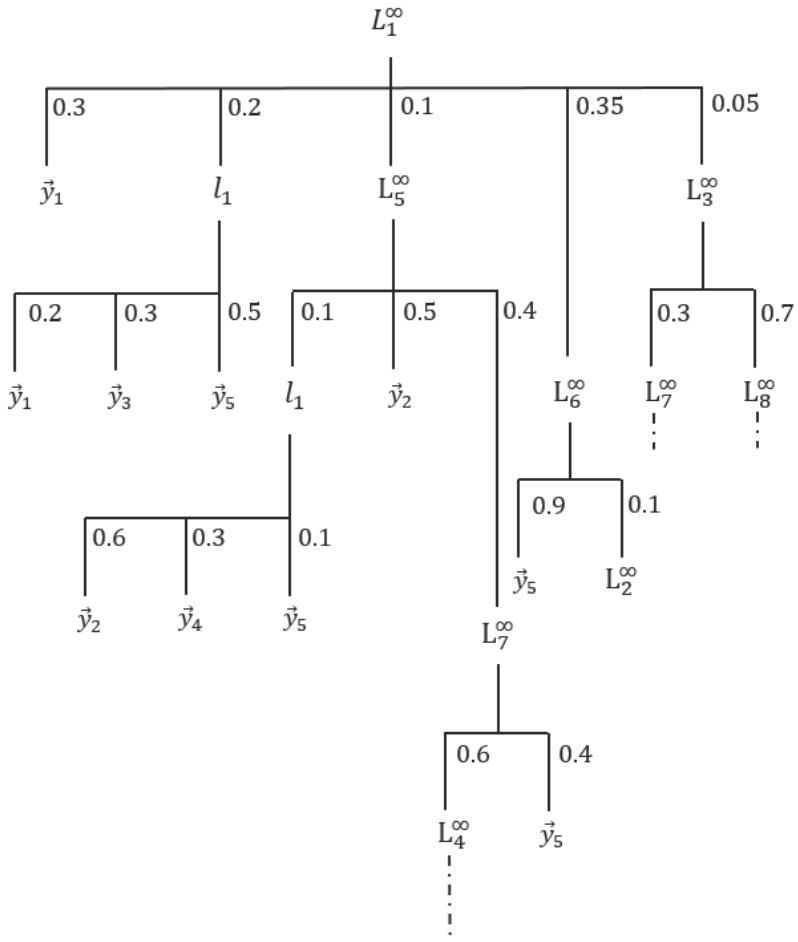


Figure 2-4: Example of an ICOL

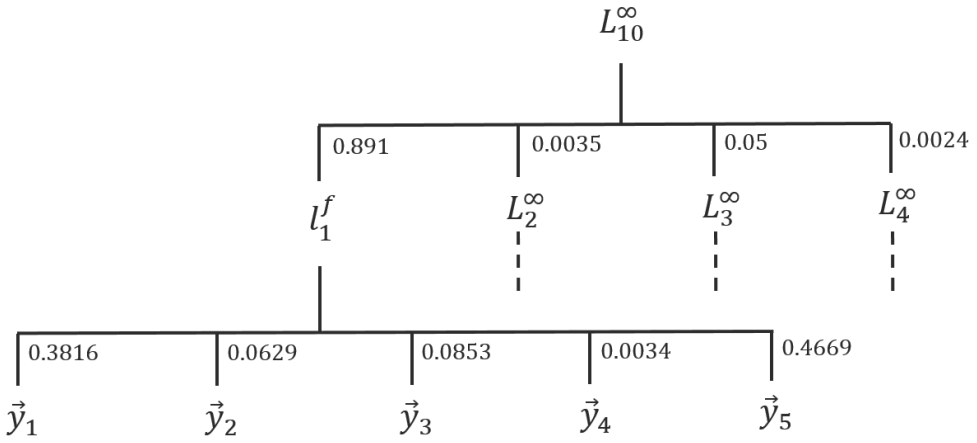


Figure 2-5: A Reduction of the ICOL from Figure 2-1

To define the probabilities in l_1^f, the aggregated probabilities must be normalized by their sum of 0.891. Then, $p(\vec{y}_1) = 0.3816$, $p(\vec{y}_2) = 0.0629$, $p(\vec{y}_3) = 0.0853$, $p(\vec{y}_4) = 0.0034$, and $p(\vec{y}_5) = 0.4669$.

The lottery l_1^f sits in the maximum reduced ICOL L_{10}^{∞} with a probability of 0.891.

Similar logic is applied to derive the probabilities of the ICOLs in the direct outcomes of L_{10}^{∞}: $p(L_2^{\infty}) = 0.035$; $p(L_3^{\infty}) = 0.05$; $p(L_4^{\infty}) = 0.024$.

The resulting lottery $L_{10}^{\infty} \sim L_1^{\infty}$ since the possible outcomes and their aggregated probabilities are the same, hence any calculated expected utility will be equal. The lottery L_{10}^{∞} from Figure 2-2 is a maximum reduced version of the lottery L_1^{∞} from Figure 2-1. Therefore, L_{10}^{∞} is a maximum reduced ICOL, though is not the only possible maximum reduced ICOL.

The following conditions are applied to restrict the possible ICOLs in *L* considered in the present work:

1. Every maximum reduced ICOLs in L has only a single ICOL within its direct outcomes
2. Every ICOL within the maximum reduced ICOLs from *L* also has *a single ICOL within its direct outcomes*

ICOLs meeting these conditions are called *ICOLs of first order*. The special recursive case of ICOL of first order is discussed in the next section and the remainder of the present Thesis. Discussion and evaluation of other special cases of ICOL can be found in Cullum et al. (2019). It is worthwhile attempting to reduce ICOLs in *L* to obtain one of the special cases as it is possible to simplify the task of determining an expected utility solution.

II. Recursive ICOLs

If a maximum reduced ICOL of first order L_k^∞ has itself as one direct outcome with probability p_k and another fictitious simple OL l_k^f with probability $(1-p_k)$, then the lottery is termed a *recursive ICOL*. Figure 2-6 presents the general structure of a recursive ICOL. This is the simplest special case of an ICOL as the same structure of the direct outcome repeats infinite number of times.

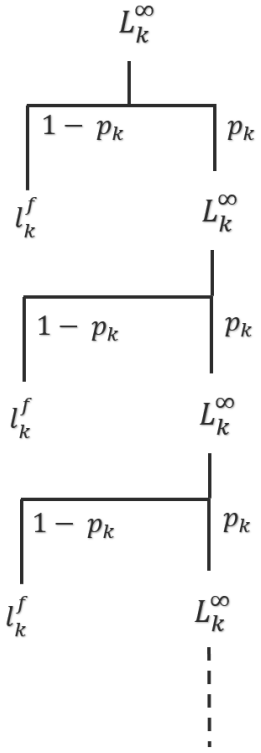


Figure 2-6: General Structure of a Recursive ICOL

The existing axioms and theorems of expected utility and Figure 2-6 can be used to determine:

$$E\left(u \mid L_k^{\infty}\right) = p_k E\left(u \mid L_k^{\infty}\right) + \left(1 - p_k\right) E\left(u \mid l_k^{f}\right)$$

Which can be simplified to give the expected utility solution (2-22) below:

$$E\left(u \mid L_k^{\infty}\right) = E\left(u \mid l_k^{f}\right) \tag{2-22}$$

The expected utility of l_k^f can be easily calculated as this is a simple OL. It can be concluded that for recursive ICOLs there exists an analytical solution for their expected utility, which equals the expected utility of the fictitious simple OL. Further, this relates the original ICOL and fictitious simple OL part:

$$L_k^{\infty} \sim l_k^{f}$$

The purpose of reducing compound lotteries is to find a simple lottery, which is equally preferred to the compound one. The fictitious simple OL l_k^f in the recursive case is a fully reduced version of the recursive ICOL L_k^{∞}.This theory is demonstrated using the following backgammon example.

III. The Backgammon Die Roll

A maximum reduced recursive ICOL may be used to model the uncertain outcomes resulting from a roll of a fair six-sided die. The possible outcomes of this ICOL include landing on each of the faces and the cocked position. In the game of backgammon, rolling the cocked position necessitates that the player re-rolls the die before he can proceed with the game. In this example, it is assumed that the DM retains his preferences for rolling each number and the cocked position and therefore that his utilities are the same for each roll of the die. Otherwise, the ICOL is not a recursive maximum reduced ICOL.

A recursive maximum reduced ICOL describing the rolling of a six-sided fair die is shown in Figure 2-7. The possible outcomes of the roll, i.e. the die landing with numbers 1 - 6 face up are described as y_1 to y_6 respectively, while the cocked position is represented as C.

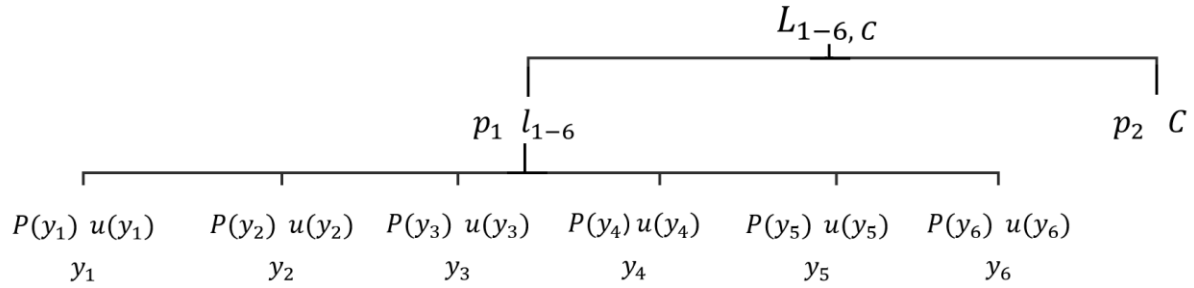


Figure 2-7: Six-sided Die Roll as Lottery

To rank the lotteries within the overall ICOL shown in Figure 2-7, the utility of *C* must be determined. *C* describes the pseudo-consequence of a never-ending game as the backgammon player is required to immediately re-roll the die. Applying the new theory, the expected utility of C must equal the maximum expected utility of the overall lottery $L_{1-6,\,C}$. *Let* $E(u \mid C) = z$.

Considering Figure 2-7; if $u(y_1)$ to $u(y_6)$ are elicited from the backgammon player as 0.1, 0.2, 0.15, 0.25, 0.05 and 0.25; and it is determined that the probabilities $P(y_1)$ to $P(y_6)$ are 0.4, 0.005, 0.01, 0.1, 0.02 and 0.465; then current expected utility theory may be applied to show that $E(u \mid l_{1-6}) = 0.18475$.

Then $E(u \mid L_{1-6,C})$ may be expressed as:

$$E(u \mid L_{1-6,\,C}) = p_1\ l_{1-6} + p_2\ C$$

Substituting in *z* and the value of $E(u \mid l_{1-6})$:

$$z = p_1\, 0.18475 + p_2\ z$$

And if $p_1 = 0.99$ and $p_2 = 0.01$, then:

$$z = \frac{0.182903}{0.99} = 0.18475 = E(u \mid l_{1-6}) = E(u \mid L_{1-6,\,C})$$

The same value of z may be obtained given that C represents an infinite geometric series. Figure 2-8 illustrates the general structure of $L_{1-6,C}$.

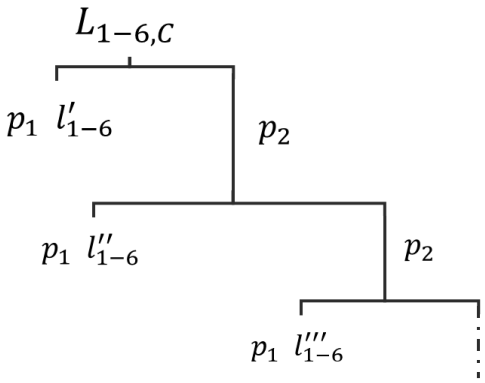


Figure 2-8: Structure Represented by $L_{1-6,\,C}$

l'_{1-6}, l''_{1-6}, and l'''_{1-6} represent the utilities placed on the outcomes of the future rolls of the die. In this example, utilities were assumed to be constant, meaning: $l_{1-6} = l'_{1-6} = l''_{1-6} = l'''_{1-6} = 0.18475$.

$E(u \mid L_{1-6,C})$ may be expressed as the following infinite geometric series:

$$E\left(u \mid L_{1-6,\,C}\right) = p_1\, E(u \mid l_{1-6}) + p_2\, p_1\, E(u \mid l_{1-6}) + p_2^{\,2}\, p_1\, E(u \mid l_{1-6}) + p_2^{\,2}\, p_1\, E(u \mid l_{1-6}) + \ldots.$$

And the expected utility solution determined as:

$$E(u \mid L_{1-6,\,C}) = z = \frac{p_1\, l_{1-6}}{1 - p_2} = \frac{0.99 \times 0.18475}{1 - 0.01} = 0.18475 = E(u \mid l_{1-6})$$

Chapter 4 develops ICOLs for the pump maintenance application and derives solutions for the special recursive case of maximum reduced ICOL.

2.2.5.2. Utility and Multi-attribute Utility Theory (MAUT)

Utility is a measure of a DMs preference for, or value which they place on a given consequence of their decision (Keeney & Raiffa, 1993). A utility function can be defined within the range $u(.) \in [0;1]$ which corresponds to a DM's increasing preference toward an occurrence.

Accordingly, a utility function may be used to determine and compare multiple utility values and can be used to rank different outcomes in alignment with the DM's preferences. This ranking must be considered separately from decision-making as preferential ranking only assumes certainty as discussed in Section 2.2.2.1.

Multi-attribute Utility Theory (MAUT) can be applied to value a prize or outcome of a lottery considering multiple factors or attributes simultaneously (French, 1986; Jansen, 2011). The attributes must be measurable, using either qualitative or quantitative means.

Initially, attribute utility functions may be obtained from the DM using utility elicitation methods. These are discussed in the following subsection. Then attribute utility functions are combined according to the sum of their scaling constants to produce the multi-attribute utility (MAU) function (Nikolova et al., 2008a; Nikolova et al., 2008b).

Scaling constants which sum to one result in an additive function. Otherwise a multiplicative function is obtained. Generally, the multiplicative form is a fair assumption for cases of more than two attributes, as the determination of scaling constants may be time consuming for larger problems. In the present work, the scaling constants were elicited directly from the DM, although alternative methods to obtain the attribute weights have been used in MAUT applications (Jansen, 2011).

I. Measuring Utility

Direct Utility Elicitation (DUE) methods are used to obtain individual attribute utility ranges. DUE methods have been used in MAUT studies in various fields, including maintenance scheduling (de Almeida, 2007; de Melo Brito et al., 2010; Ferreira et al., 2009; Karydas & Gifun, 2006; Schulz & Stehfest, 1984; Teixeira de Almeida, 2001).

DUE methods involve a structured interview process where the DM is guided using a reference experiment. These methods draw on Keeney and Raiffa (1993), and include Probability Equivalence, Certainty Equivalence, Lottery Equivalence and Uncertainty Equivalence (Tenekedjiev, 2007b). The simplest method possible should be selected to ensure the best

possible result. Otherwise, if the DM has difficulty determining his preference for uncertain outcomes, the analyst may not be able to accurately model the DMs preference structure using methods such as Lottery Equivalence and Uncertainty Equivalence. Some additional DUE variations and alternatives exist (Heins & Röling, 1995; Ulusoy et al., 1992) .

Certainty Equivalence and Probability Equivalence are the simplest of these methods. These are applied in Chapter 4 to elicit utility range values from the expert in the development of the pump RBM system.

Considering the Certainty Equivalence method, DM's indifference toward some value of probability p versus the outcome is used to determine utilities, as per the preferential equation shown in (2-23) which is outlined in texts such as French (1986).

$$\langle y_{best}\ p\ y_{worst} \rangle \sim A \tag{2-23}$$

In (2-23), y_{best} is the best available consequence, and y_{worst} the worst consequence of the decision according to the DM. A is the current outcome or alternative being considered, and p is the solution variable, which can be shown to be equal to the utility of the alternative (French, 1986) at the point of indifference. The solution variable is represented by a reference experiment during elicitation to aid his comprehension.

The reference experiment is a graphical or hypothetical construction which describes two alternatives weighed against one another. A 'urn of balls' reference experiment may be used which consists of a hypothetical urn of 100 identical balls in total. The balls differ only in colour. Some proportion of these are black and some are white, and the proportion is known to the participant and analyst. The analyst assigns either the black or the white portion to represent p. Questioning proceeds and the value of p is adjusted until the DM is indifferent between the alterative A and p.

While DUE methods extract point values, real DMs are 'fuzzy' rational, incapable of infinite discrimination of their preferences (Nikolova et al., 2006). Therefore, a utility range must be

elicited in preference to point values. DUE methods can be applied to determine the upper and lower limits of this range, although this is a time-consuming process due to the large amount of values and ranges. The number of utility ranges required and the number required to model a continuous utility function may be reduced with the incorporation of the Triple Bisection Method (Tenekedjiev, 2007a).

Determination of the utility ranges allows for the creation of a utility function when these are fitted to a mathematical function. The gradient of the function can also be used to determine the DM's local risk attitude. The 'arctan' model produces an accurate model of the utility function (Nikolova et al., 2009), and is used to develop attribute utility functions in Chapter 4.

I. Measuring Scaling Constants

Scaling constants for the MAU function represent the importance of each attribute individually to the DM. Scaling constants are elicited in a similar way to the Certainty Equivalence method. A preferential equation is used to find the point of indifference between a reference gamble and extreme values of each attribute (Clemen, 1996). These scenarios are also known as corner consequences. The vector $y_{1,\,Corner}$ in (2-24) gives an example of a corner consequence wherein one attribute is best, and all others are the worst possible.

$$y_{1,\,Corner} = \left[y_{1,\,Best},\, y_{2,\,Worst},\, y_{3,\,Worst} \ldots y_{n,\,Worst} \right] \tag{2-24}$$

Following the fitting of the utility attribute functions and elicitation of their scaling constants, the MAU function may be created and evaluated to produce utility values for expected utility calculations.

2.2.5.3. Subjective Probability

Subjective probability can be defined as an observers' unique perception of the probability of an event occurring. This perception is based on their knowledge and experience relating to the event. It is necessary to obtain subjective probability values when equivalent values cannot be measured

for the evaluation of a lottery model. Ideally, subjective probabilities should be obtained from the DM as these values will correspond with their utilities. If it is not possible to survey the DM, a suitable expert should be surveyed to obtain subjective probabilities.

I. Measuring Subjective Probability

Subjective probability assessments should be guided for maximum comprehension of the method and results by the participant and the analyst (Jaspersen & Montibeller, 2015). Guidance is provided using a reference experiment which may be similar to that used for DUE methods described in Section 2.2.5.2. Visual aids include probability scales, gamble-like methods, and probability wheels. Renooij (2001) evaluated these methods and found advantages and disadvantages to each, though no optimal method was suggested. However, Haase et al. (2013) suggested that numerical approaches are the most accurate. Therefore, the DM may be asked directly to estimate the probability of an event occurring as represented by a numerical scale, gamble or wheel if he is not already certain of the percentage chance. A comprehension advantage is provided by scales and wheels due to their simplicity.

Given that DMs are considered 'fuzzy' rational and utility and subjective probability elicitation methods are similar, it is reasonable to expect that ranges of values are obtained from these assessments. However, discrete values result as the DM can comprehend probabilities more easily than his preferences, and the analyst can give the DM confidence in his assessment of a discrete value using an appropriate visual aid.

2.2.5.4. Evaluating Expected Utility for a Decision Tree

Once relevant probability and point MAU values have been obtained, the expected utility of all lotteries within the decision tree can be evaluated.

There are many types of lotteries, and each type of lottery requires a different method of solution. Simple OLs and FCOLs were discussed in Section 2.2.5.1, followed the development of a reduction algorithm and solution method for the special recursive case of ICOL. In the latter case,

it was necessary to determine the expected utility of the fictitious simple OL. The present section describes the expected utility solution of a simple OL.

Recall a simple ordinary lottery directly results in discrete outcomes or consequences following one draw. The expected utility $E(u|l_s)$ of a simple lottery for the present maintenance application is calculated using (2-25) which is developed from the general expression in French (1986).

$$E(u|l_s) = \sum_{1}^{j} P(\theta_j | \vec{x}) \times U(\vec{y}_i) \tag{2-25}$$

In the present application, probabilities of 'no fault found' or 'fault found' states in equipment are represented by $P(\theta_j | \vec{x})$, which are conditional given some measurement $\vec{x}$, and the value of each consequence may be may be measured according to its MAU value $U(\vec{y}_i)$. As discussed previously, a suitably trained linear discriminant classifier can produce $P(\theta_j | \vec{x})$ for measurable events, and an MAU function may be developed by eliciting utility and scaling constant values from the relevant DM. MAU function development is described in further detail in Chapter 4.

Ultimately, the expected utility of all ordinary lotteries within a decision tree can be determined as a number between zero and 1, with the highest number corresponding to preferred action of the DM. Accordingly, this overall maximum value for the decision tree can calculated by the maintenance system as the maintenance Policy.

2.2.5.5. Decision Trees

Decisions under strict uncertainty or risk may be modelled as decision trees to represent the connections between action and consequence. This type of decision tree is unique to this field and is not to be confused with a 'tree' flowchart (Woitek et al., 2017) or 'decision tree' classification

method (Kingsford & Salzberg, 2008; Kotsiantis, 2013). It is a third type which can model uncertainty using lotteries at the tips of its branches, whereas the others cannot.

A decision tree for a decision under risk such as a maintenance context may be created by considering that its branches eventually terminate in uncertain actions, which in this case either represent a maintenance action or a deferral action. Such a decision tree shall possess a decision node at its root, which branches toward each action. Branches may divide at chance nodes due to uncertain circumstances, which can be modelled as a lottery. Branches terminate when they reach the maintenance actions. These involve uncertain outcomes and can also be modelled as lotteries.

Decision nodes, chance nodes and actions can be illustrated as circles and squares respectively, connected by lines which represent the branch. The structure of the tree is specific to the decision problem and must be developed accordingly.

Usually trees are solved backwards from the consequences by computing expected utilities at the lottery nodes at end of each branch, then at each node or fork in a branch, and lastly by eliminating branches which do not possess the maximum expected utility until all branches are solved. The branch which results in the maximum expected utility is called the Policy. A Policy can represent one or more real actions depending on the structure of the decision tree. In any case, the expected utility of all nodes must be computed to determine the Policy.

2.3. Conclusions

From the breadth of literature discussed previously, the following conclusions can be drawn:

Section 2.1.1 presented ten examples of previous studies which used a combination of CM and ML to perform fault detection of marine equipment without further maintenance scheduling or decision-making. Vibration, temperature and pressure measurements were often used to parameterize the equipment faults, while supervised ANNs or SVMs were used to determine the highest probability fault occurring or predict the future performance of the system. ANNs or SVMs were implemented as either classification or regression algorithms. Only classification algorithms could be used to identify multiple faults, which was defined as a previous RBM

system requirement in Chapter 1. Some of these authors emphasised that it was difficult or not possible to obtain enough quality data for their models which indicated clearly that each fault was occurring. This reduced the accuracy of the models as: it is difficult to determine which measurements indicate the symptoms or presence of each fault; certain algorithms require large datasets for acceptable performance and it is expensive to obtain the necessary data. Section 2.1.3 concludes that the minimal amount of measurements must be used, and that these should contain as much fault information as possible to ensure enough high-quality data are be obtained within resource limitations. Vibration Analysis is suggested based on its use in existing studies. Secondly, Section 2.1.3 concludes that the classification algorithm with the highest estimated performance on the smallest possible dataset should be chosen to minimise application cost. Following a discussion of performance estimation methods for these algorithms, Section 2.1.3.1 concludes that there is no universally accepted method except for an inefficient interpretation of the confusion matrix.

Supervised classification algorithms are commonly used to classify or produce a class label based on the most probable fault instead of the probabilities of all faults. Supervised Bayesian algorithms can be used to produce probabilities, while non-Bayesian approaches can be used to approximate probabilities. Section 2.1.3.1 also discusses how calculation of all fault probabilities can be useful in some cases such as fault detection. As described in Chapter 1, calculation of all fault probabilities fully quantifies the risk involved in operation and maintenance of the equipment. This is necessary for the development of the RBM system in the present Thesis.

Although it is possible to produce the probabilities of all possible faults using Bayesian algorithms, there was no widely known method used to meaningfully interpret the probabilities prior to the work conducted in the present Thesis. Accordingly, the remainder of Section 2.1.3.1 reintroduced the certainty matrix and defined the novel doubt matrix for Bayesian algorithms as tools for this purpose. Approximate matrices were also defined for non-Bayesian algorithms. The certainty matrix shows the confidence of the classifier in making correct classifications. The

doubt matrix shows the doubt of the classifier in making correct classifications. These matrices or their approximates can be used together with the confusion matrix to estimate the performance of future supervised classification algorithms. The tools are applied to estimate the performance of the classifiers built for the pump RBM system in Chapter 4.

To complete the system development discussion, Section 2.2 presented seven examples of existing studies which focused on maintenance decision-making and maintenance scheduling. The studies used MAUT, AHP, multi-objective optimisation, a combination of these methods or other methods to make decisions from equipment condition data or expert data. They commonly involved many experts in preference to a single decision-maker and produced a single result except in the case of the multi-objective optimisation techniques which produced multiple results. These existing approaches are either: very complex and make decisions based on the most probable fault only or cannot produce a single result when multiple faults are considered.

Section 2.2.2 concluded that future applications should select a simple method to reduce development time. Section 2.2.2.1 highlighted that MAUT may be beneficial but must be adapted to consider the uncertainty or risk involved in equipment operation and maintenance. Additionally, Sections 2.2.2, 2.2.3 and 2.2.4 emphasized that future systems should focus on a single decision maker and should produce a single outcome to ensure results are reproducible.

Chapter 1 suggested a Decision Theory as an overall decision-making technique, and the key principles behind this theory are presented in Section 2.2.5. Decision Theory also incorporates MAUT, adapting this approach to consider uncertainty. Decision Theory describes uncertain outcomes or decision consequences using lotteries. Lotteries can be used to model situations such as games of chance, the future price of stocks and equipment condition.

The prizes within these lotteries have some probability of occurrence and some value to the DM, which is represented as his utility. The discussion in Section 2.2.5.1 is restricted to simple lotteries which can be resolved after one draw and compound lotteries which can be resolved following multiple draws. The discussion then focuses on ICOLs, which are a special case of

compound lottery that can only be resolved after an infinite number of draws. ICOLs arise in situations such as maintenance where the consequences of a deferral action extend indefinitely into the future. Decision trees used in the pump RBM system could include ICOLs provided that their solutions could be determined.

Only approximate solutions to ICOLs could have been found prior to the work conducted in the present Thesis. A novel simplification algorithm to produce maximum reduced ICOLs and a novel solution method for recursive maximum reduced ICOLs were presented and demonstrated in the remainder of Section 2.2.5.1. Thus, ICOLs can be included in future studies which involve decision trees. ICOLs are included in the pump RBM system and are in Chapter 4. The remaining Sections 2.2.5.2 to 2.2.5.5 described how expert data in the form of utilities and subjective probabilities may be obtained to evaluate the lotteries within the decision trees.

Techniques used to classify data and make maintenance decisions have been evaluated in Chapter 2. An RBM system should be built using a supervised Bayesian classification algorithm which performs well on few data and produces probabilities. Developing supervised Bayesian classifiers using MVA was discussed in Section 2.1.3.2. The classification algorithm should be paired with decision trees containing ICOLs to analyse all probabilities and make single-outcome decisions. Lastly, re-iterating from Chapter 1, the system should be developed for an individual piece of equipment initially, before extending the scope of the application to sub-systems and systems.

2.4. Summary

Chapter 2 presented the theory and an overview of relevant studies applying CM, ML, MCDM and MAUT techniques to the maintenance of maritime applications or vessels. The need for a quality yet cost-effective data collection regime was highlighted to enable CM and ML applications for the maintenance of marine vessels. Existing applications of CM and ML demonstrated the use of vibration monitoring to produce quality and cost-effective datasets.

The issue of selecting a suitable multi-class supervised classification algorithm remained. As selection of an algorithm is commonly aided by performance estimation methods, an overview of

these was provided. Complete risk-quantification could be achieved for an RBM system if it could produce the probabilities of all equipment faults. Supervised Bayesian classification algorithms can be used to calculate probabilities, though no tools existed previously to interpret these. From this discussion, it was suggested that that the certainty matrix as defined originally in (Nedev & Tenekedjiev, 1994), a novel doubt matrix and the commonly used confusion matrix are used together as a set of performance estimation tools to interpret probabilities.

MVA can be used to create Bayesian linear discriminant classifiers with good performance on small datasets. The discriminant classifiers will interpret CM data and generate all probabilities in the present pump RBM system as the Risk Assessment element.

Considering the limitations of existing studies, Decision Theory and MAUT were suggested as a suitable combination of techniques which can be applied for rational maintenance decision-making within the Maintenance Scheduling element of an RBM system. Maintenance deferral actions can be modelled using ICOLs. A reduction algorithm and solution method were presented for the special case of recursive ICOL. Methods to obtain utility intervals and subjective probabilities were described, concluding with Section 2.2.5.5 which described decision trees.

It is anticipated that the combination of linear discriminant classifiers and decision trees containing recursive ICOLs can be applied successfully to create a maintenance system for a piece of equipment. The resulting RBM system shall interpret the current condition of the equipment into the best possible maintenance action. An RBM system is developed for a shipboard pump in the following Chapters 3 and 4. The results of various analyses using the system are discussed in Chapter 5.

CHAPTER 3 - Maintenance System Development Part I:

Stage 1 - User Prompt and Stage 2 - Data Collection and Processing

The present Thesis develops an RBM system for a shipboard pump in the present Chapter 3 and the following Chapter 4. An RBM framework was outlined in Chapter 1. CM, ML and Decision Theory were selected to form this system following discussions in Chapter 2. The present Chapter initially describes the vessel and shipboard pump application. Then, a four-stage system workflow is outlined which guides system development and the completed system's use. The first two stages are described in the present Chapter 3. These include User Prompt as well as Data Collection and Processing. The remaining two stages are described in the following Chapter 4 as well as calculations to measure system performance against periodic PM. Data processing software is attached as part of system software in Appendix C. Examples of raw data are attached as Appendix E. The results of the performance measurement are presented and discussed in Chapter 5. A summary concludes the present Chapter.

3.1. Application

The application of the maintenance system concept was facilitated by the sponsoring industry partner. This support overcame the vessel access barrier highlighted in Chapter 1.

3.2. Vessel

The application of the present methodology was carried out on a 25m tug which is normally operated in Sydney Harbour and owned by the Royal Australian Navy (RAN). She provides a critical operational function to Navy as she is responsible for moving larger ships to and from berth. Her operational profile is highly irregular, increasing or decreasing as required. She may operate independently or in collaboration with other vessels such as her near-identical sister ship. The vessel is pictured in Figure 3-1. All maintenance aboard the vessel is scheduled according to periodic PM principles or using the experience of the Chief Engineer. Routine maintenance is conducted by the Chief Engineer.

Figure 3-1: Tug Study Vessel

3.3. Number 2 General Service Pump

The component application as recommended in Chapter 1 was the Number 2 General Service Pump. This horizontal centrifugal pump is part of the bilge and fire system aboard the vessel. Reasons behind the selection of the pump were presented in Section 1.6. Figure 3-2 shows the bilge and fire system and emphasises the location of the Number 2 General Service Pump within the system. Figure 3-3 shows the Number 2 General Service Pump.

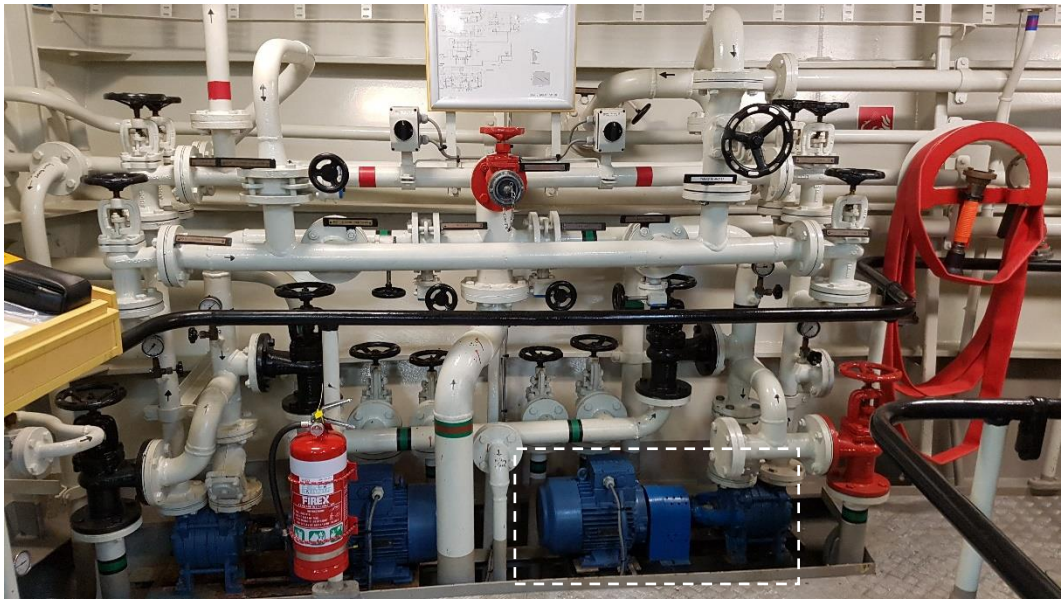


Figure 3-2: Bilge and Fire System with Emphasis on Number 2 General Service Pump

Figure 3-3: Number 2 General Service Pump

The Number 2 General Service Pump is a horizontal centrifugal pump and motor set responsible for pumping water as required in emergency scenarios, such as in the event of a fire aboard the ship. As such, its operational profile is irregular and the pump must be available on demand. The Number 2 General Service Pump may also be operated in conjunction with the Number 1 General Service Pump, although the Number 1 General Service Pump is generally used in preference. As a result, the Number 2 pump has few operational hours and generally requires little maintenance. Its deterioration occurs mainly due to infrequent use. It was estimated that the Number 2 General Service Pump has operated for 100 hours out of a possible 8100 hours since commissioning of the vessel in 2009.

3.3.1. Specifications of Number 2 General Service Pump

The 'Number 2 General Service Pump' refers to the combination of the pump body and the motor driving the pump shown in Figure 3-3. Relevant specifications of each of these are tabulated in Table 3-1 and Table 3-2 respectively.

Table 3-1: Number 2 General Service Pump Body Specifications

PUMP BODY SPECIFICATIONS	
Manufacturer and Model	Sterling Fluid Systems, AKHA5101AA0012H4
Working Fluid(s)	Seawater and Fresh Water
Nominal Flow Rate	$20m^3/hr$
Main Materials	Cast Iron (Casing), Bronze (Impeller)
Vanes in Impeller	21
Balls in Drive End Bearing	8

Table 3-2: Number 2 General Service Pump Motor Specifications

MOTOR SPECIFICATIONS	
Manufacturer and Model	Rotor, 5RN132S04
Rating	6.3kW, 11.2A, 460V (Delta), IP 55
Speed	1750RPM
Frequency	60Hz

3.3.2. Maintenance of the Number 2 General Service Pump

The periodic PM schedule for the pump is outlined below in Table 3-3.

Table 3-3: Periodic PM for Number 2 General Service Pump

Schedule	Tasks
Biannual	Change non-drive end bearing grease
Annual	Change drive-end bearing grease
Month	Check drive-end bearing grease and top-up as required
Week	Visual inspection of pump and motor

Corrective maintenance activities were carried out on the Number 2 general Service Pump in March 2015. The packing in the pump was replaced, pump and motor were realigned, and the base of the pump was painted. Since March 2015, the only maintenance performed on this pump was a periodic inspection of the grease in the drive-end bearing of the pump and re-greasing as required. Re-greasing the bearing was necessary approximately every three months. It is assumed that the Number 2 General Service Pump was free of any known faults at the commencement of the CM data collection period. The maintenance history of the Number 2 General Service Pump since March 2015 to the conclusion of the CM data collection period is attached as Appendix B.

RCM was also discussed in Section 1.1 as a common maintenance approach used for marine applications. However, it shall not be discussed further as it was determined to be sub-optimal and has not been applied to the present case study.

3.4. System Workflow and Development Overview

Operation of a completed RBM system can be broken down into a four-stage cycle: User Prompt, Data Collection and Processing, Data Analysis and Storage and Transmission. This is shown in in Figure 3-4. The purpose of each of the stages and the development of pump software for the first two stages is described in the remainder of the present Chapter. The remaining two stages are descried in Chapter 4. The pump software was developed in MATLAB R2018b in combination with the Statistics and Machine Learning Toolbox (The MathWorks Inc, 2018) and is attached as

Appendix C. While MATLAB was used to develop the present software, future applications are not restricted in their choice of scripting language.

1. **User Prompt:** In the first part of the cycle, software prompts the user to select the context of the decision. The context is a pre-programmed description of the operational environment of the application. The operational environment may describe the status of a larger system or equipment in the vicinity. This context determines which maintenance actions are reasonable, and therefore which decision models which are evaluated using the system.
2. **Data Collection and Processing:** Experimental and CM data are required to develop system software in the following Stage 3. Initially, labelled experimental data required for the training of the classifier or alternative method are collected. These data must distinguish each 'fault found' or 'no fault found' condition. 'No fault found' conditions must include the typical routine operation of the equipment. If appropriate data are available, no additional experimental work is required. Appropriate CM data must also be sampled from the specific application during its normal operation for as long as practical, with the minimum being two occurrences of maintenance work. Data Collection and Processing may be achieved manually or automatically with the installation of automated sensors and appropriate software. Data Collection and Processing should be completely automated where practical.
3. **Data Analysis:** Experimental data are used to develop the discriminant classifier(s) or an alternative to generate probabilities. The principles of LDA for classification and the development decision trees using Decision Theory were described previously in Chapter 2. Expert guidance and expert data are then used to develop the appropriate decision tree or trees. The classifier(s) and decision tree(s) are combined to form the completed system. In Stage 3, the completed system has been provided with the decision context, prompts the user to input current data or automatically collects it and then selects the best possible maintenance action. Display or output of the probabilities is not necessary as the decision trees incorporate all of

these to calculate the best possible decision. The output of the maintenance system in this stage is a list of expected utilities for each action and lastly, the maintenance Policy.

4. **Storage or Transmission:** The Policy, expected utilities and time and date information are stored or transmitted to the organisation's maintenance management system or similar until these data are no longer required. Following the successful Integration phase of an application described in Chapter 1, it is suggested that reception of this data triggers either a message for a maintainer near the equipment or the generation of a work order within an organisations' Computerised Maintenance Management System (CMMS) if maintenance is required.

At the end of each cycle, the performance of the overall maintenance strategy using availability and overall maintenance cost can be assessed as described previously in Chapter 1.

It is expected that the user repeats the workflow cycle as required for the piece of the equipment. Long-term equipment condition trends are not analysed within the present work due to time and resource limitations, although it is possible to adapt the resulting system for use in a future predictive maintenance system. Predictive maintenance is discussed further in Chapter 5.

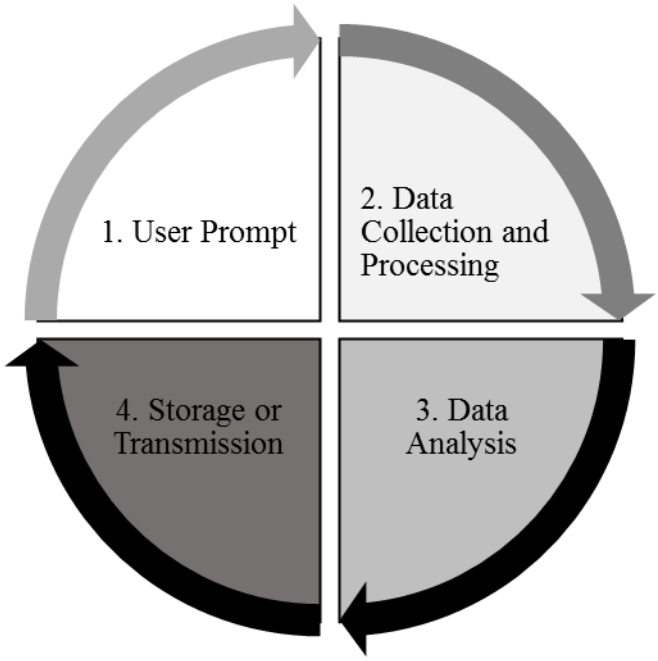


Figure 3-4: Workflow Process

The remaining Chapters of this Thesis describe the development and evaluation of the pump RBM system created according to this four-stage process.

3.5. Stage 1: *User Prompt*

As part of an overall main script file used for Data Analysis, a function may be created in MATLAB which requests input from the user as follows:

1. Display decision context descriptions alongside decision context numbers to user
2. Request decision context number from user and store

This variable is then used by the RBM system in Data Analysis to produce the maintenance action recommendation.

The decision context represents the operational environment of the pump in this application. In the present study, four decision contexts were considered. These contexts included: the vessel is moored at the wharf (alongside) and no other equipment is operating, the vessel is alongside and the main engines are running, the vessel is slow steaming at <4 knots in the harbor (at sea) and no emergency occurs and lastly that the vessel is at sea and it is an emergency. Implemented systems require this User Prompt stage as the decision context is guaranteed to change.

3.6. Stage 2a: *Data Collection*

This section describes the methods used to collect and process experimental and CM data for the development of the linear discriminant classifiers within the present RBM system.

3.6.1. Experimental Data Collection

Experimental data were required to develop the discriminant classifiers. Ten experiments were designed and conducted for this purpose. Each experiment simulated one of three ‘no fault found’ conditions or alternatively one of seven common centrifugal pump faults as a ‘fault found’ condition. ‘No fault found’ conditions included operation of the pump when: the vessel was alongside and the engine room was quiet, the vessel was alongside and the main engines were

running and lastly when the vessel was at sea. ‘Fault found’ conditions included a worn impeller, damaged packing, a damaged drive-end bearing, a worn drive-end bearing, a loose mounting, an unbalanced shaft and a misaligned shaft as described in literature (Beebe, 2004). Further description of these faults is provided in Section 3.6.2.

In uncommon or complex applications, fault information may not be available in literature. In this case, approaches such as Failure Modes and Effects Analysis (Cicek & Celik, 2013; Feili et al., 2013; Kahrobaee & Asgarpoor, 2011; Xu et al., 2017) have been utilized in many applications and can be used to identify faults and measurable fault symptoms. Further, significant faults in a given application may be identified using existing maintenance records.

To avoid affecting the operation of the vessel all experiments were performed using a temporary ‘test pump’ shown in Figure 3-5 and Figure 3-6.

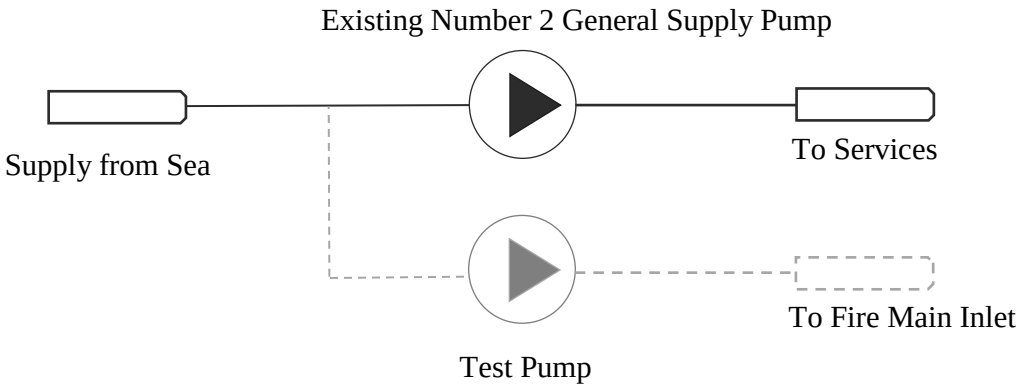


Figure 3-5: Diagram of Test Pump Configuration

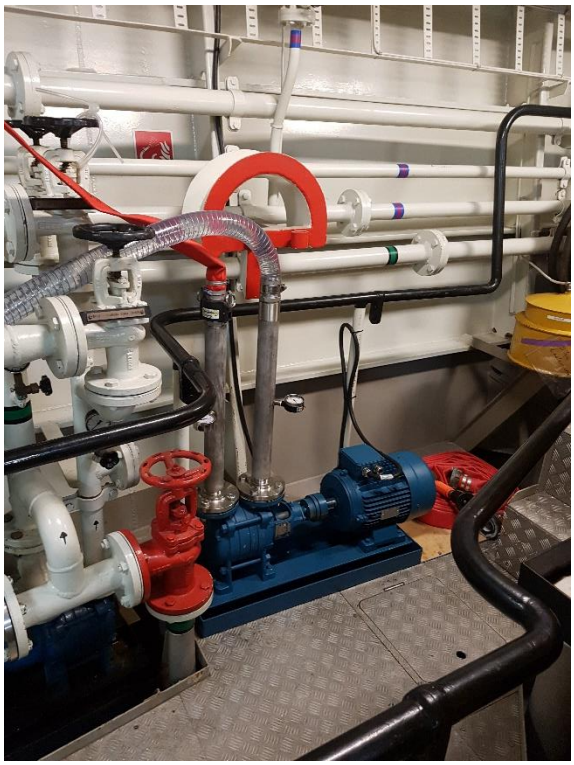


Figure 3-6: Test Pump Configuration

Each experiment was performed once for 20 minutes. The ‘normal operation while the vessel was alongside and engine room quiet’ experiment was repeated a further four times as this was the most common operational scenario. The test pump suction pressure to was set to -0.1 bar. The discharge pressure was set to 2.1 bar. These pressures resulted in a seawater flow rate of approximately $20m^3hr^{-1}$. During each of these experiments 2s vibration samples were collected at each point according to Figure 3-7 and Table 3-4, while pressure gauges were read each minute as recommended by relevant literature (Lee et al., 2014). Vibration measurement points were chosen according to guidance provided by an appropriate vibration analysis training resource (Mobius Institute, 2015). The shaft RPM was obtained once per experiment for subsequent vibration data processing.

Temperature measurements were obtained according to Figure 3-8 and Table 3-5. Temperature measurements were corrected using FLIR software for the ambient temperature and humidity of the engine room.

Additional motor current and packing drip rate measurements were obtained each minute as they also aid in distinguishing faults (Beebe, 2004), are easily obtained and were used by the Chief Engineer to monitor the pump.

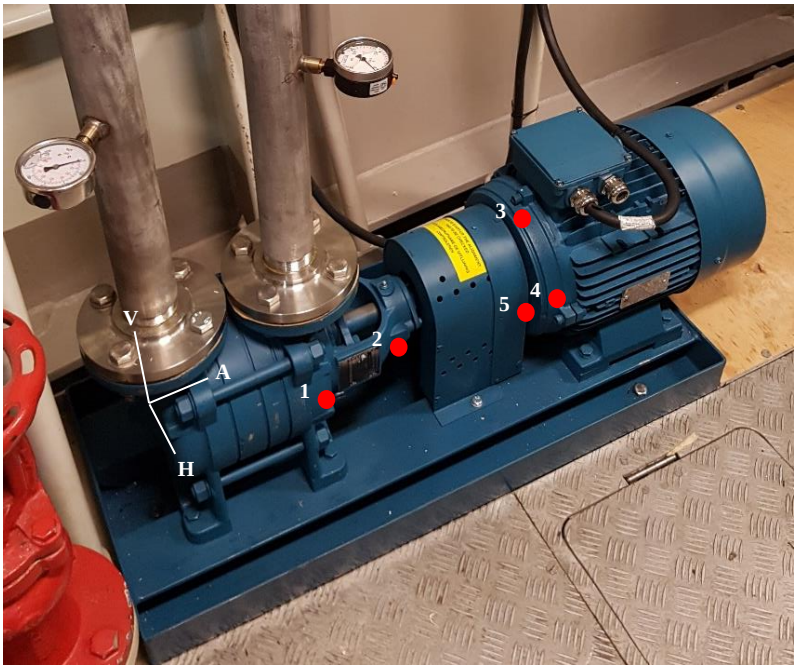


Figure 3-7: Vibration Analysis Measurement Points

Table 3-4: Description of Vibration Analysis Measurement Points

POINT	1	2	3	4	5
LOCATION	Pump, Casing	Coupling Guard	Motor, Casing	Motor, Casing	Motor, Casing
ORIENTATION	Horizontal	Horizontal	Vertical	Horizontal	Axial

The ‘Horizontal’(H), ‘Vertical’(V) and ‘Axial’(A) directions are indicated on Figure 3-7. These directions correspond to the orientations of the vibration measurement probes used to obtain each measurement.

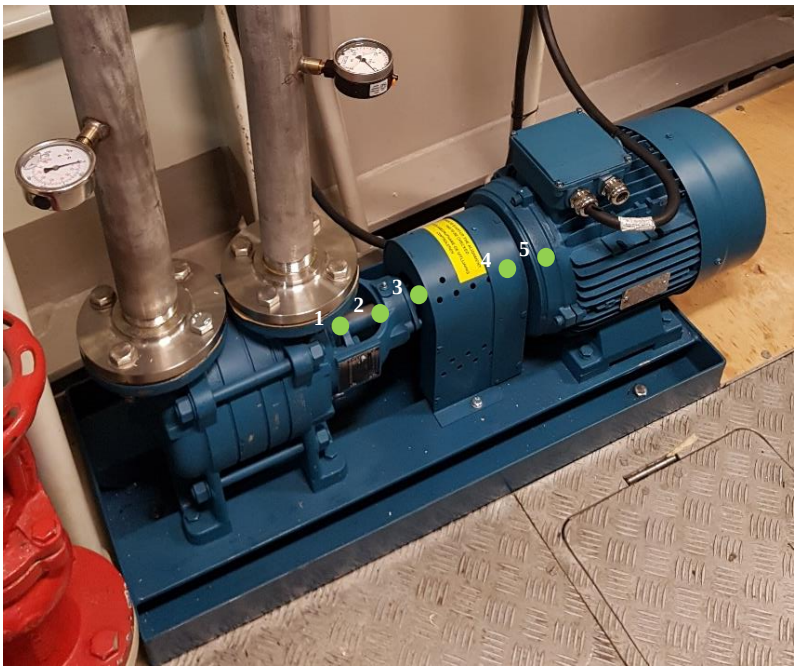


Figure 3-8: Temperature Measurement Points

Table 3-5: Description of Temperature Measurement Points

POINT	**1**	**2**	**3**	**4**	**5**
LOCATION	Pump, Bearing Casing	Shaft	Coupling	Motor, Drive End Bearing	Motor, Casing

The measuring devices used, sampling rate and number of samples obtained per experiment are listed in Table 3-6.

Table 3-6: Characteristic Measurements

MEASUREMENT	DEVICE/METHOD	SAMPLING RATE	NUMBER OF SAMPLES PER EXPERIMENT
VIBRATION, 5 POINTS ON PUMP AND MOTOR VELOCITY FFT AND VELOCITY TIME WAVEFORM	Commtest vb7 Vibration Analyser, Ascent 2015 Software for FFT	400 Hz, 1.25 Hz Bins	10 x 5 locations
SUCTION PRESSURE	Reading pressure gauge on suction, calibrated to Australian Naval standard	Per minute	10
DISCHARGE PRESSURE	Reading pressure gauge on discharge, calibrated to Australian Naval standard	Per minute	10
TEMPERATURE, 5 POINTS ON PUMP AND MOTOR	Extract from continuous recording of FLIR Thermal Imaging Camera using FLIR Tools and software	Per minute (Extract from continuous recording per point)	10
MOTOR CURRENT	Reading display of motor current clamp	Per minute	10
PACKING DRIP RATE	Counting number of drips of water from packing each minute	Per minute	10
ROTATIONAL SPEED OF SHAFT (RPM)	Tachometer and reflective tape	Per experiment	1

3.6.2. Faults

Experimental conditions were created which simulated seven ‘fault found’ conditions in the test pump, so that these could be detected in the Number 2 General Service Pump. Cavitation was excluded from the present study as this phenomenon occurs during the normal operation of the Number 2 General Service Pump and therefore would not indicate a fault.

The relationships between the measurements in Table 3-6 and the faults are shown in Table 3-7, reproduced from Beebe (2004). The fault symptoms ‘S’ can be detected using some combination of measurements in each case.

Table 3-7: Measurements and Characteristic Fault Symptoms

FAULT	FLUID LEAKAGE	DIMENSION CHANGE	POWER	HEAD OR VACUUM	FLOW	SPEED	VIBRATION	TEMP	COASTDOWN TIME	WEAR DEBRIS IN OIL	OIL LEAKAGE
DAMAGED IMPELLER		S	S	S	S	S	S	S	S		
DAMAGED EXTERNAL SEALS	S	S		S		S	S				
ERODED CASING		S									
WORN SEALING RINGS			S	S	S						
ECCENTRIC IMPELLER			S	S		S	S	S	S		
BEARING DAMAGE		S	S			S	S	S	S	S	S
BEARING WEAR		S					S	S	S	S	
MOUNTING FAULT							S	S			
UNBALANCE (SHAFT)							S				
MISALIGNMENT (SHAFT)		S					S				

The methods used to create the seven experimental ‘fault found’ conditions are described in the following sub-sections.

3.6.2.1. Impeller Wear

The clearance of the impeller was increased from an estimated 15µm to 20 µm using a lathe to remove material from the fluid-entry side. The fluid-entry side was then polished to round the edges of the machined surface as shown in Figure 3-9 simulating wear of the impeller and exposing the bronze beneath the protective coating. The duration and resources available to the present study prevented the testing of multiple wear states or an accelerated wear testing approach.

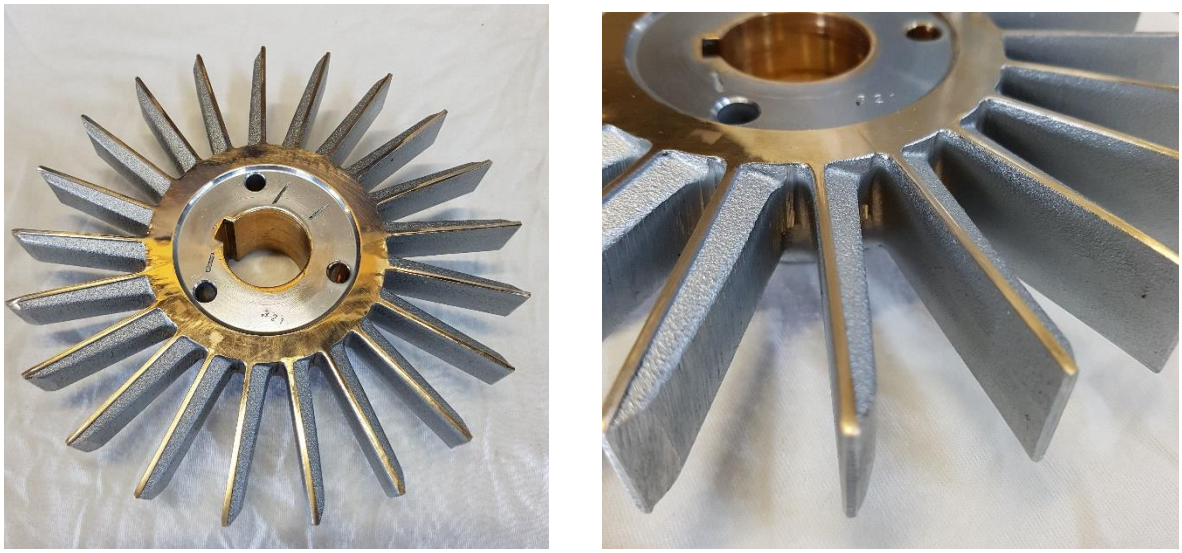


Figure 3-9: Impeller Wear Fault Simulation

This approach ensured material was removed to simulate wear of the impeller. However, this did not compromise the integrity of the impeller and introduce a safety risk unlike alternative approaches such as cutting or damaging vanes.

3.6.2.2. Damaged Pump Packing

The packing ring between the two halves of the pump casing was loosened to simulate damage over time by loosening the casing retaining bolts to finger tightness. This caused the pump to leak at a higher rate than observed when the bolts were tightened with a spanner.

3.6.2.3. Damaged Pump Drive-End Bearing

A grinding wheel was used to remove 75% of the thickness of a section of the bearing's outer race to simulate a bearing requiring immediate replacement. The thickness reduction is emphasised by the shadow shown on the top surface of the bearing in Figure 3-10. Subsequently, the surface and its edges were polished for a smooth finish. This approach was preferred to existing alternatives (Kamiel, 2015; Sawalhi et al., 2007) to minimise stress concentrations in the bearing caused by sharp edges and the likelihood of complete bearing failure during the experiment. Safe operation of the pump was ensured by a thorough greasing of the bearing when it was reinstalled.

Figure 3-10: Damaged Bearing Fault Simulation

3.6.2.4. Worn Pump Drive-End Bearing

A similar pump which was part of the air conditioning unit was monitored to investigate the effects of a worn bearing. The bearing in the pump was expected to show more evidence of wear as it had seen more operational hours than the Number 2 General Service Pump. The duration and resources available to the present study prevented an accelerated wear testing approach using the test pump, or the ability to model more than one wear state in the bearing. The air conditioning pump shown in Figure 3-11 has similar specifications, though it has a mechanical seal in place of a packing ring to minimise the effects of corrosion during its more frequent operation.

Figure 3-11: Air Conditioning Pump with Mechanical Seal

Using the air conditioning pump in place of the test pump meant that the suction pressure was raised to 0.1 bar to prevent damage to the air conditioning system. Using the air conditioning pump also means that it was not possible to measure the packing drip rate. It is expected that the results of the experiment are affected by the difference in pump configurations.

3.6.2.5. Loose Mounting of Pump Foot

One of the test pump mounting bolts was loosened from its usual tightness by a half-turn with a spanner to replicate loose mounting. This is highlighted in Figure 3-12 as the rear left-hand bolt on the pump. The remaining three bolts were not loosened, which ensured the safest configuration for personnel conducting measurements.

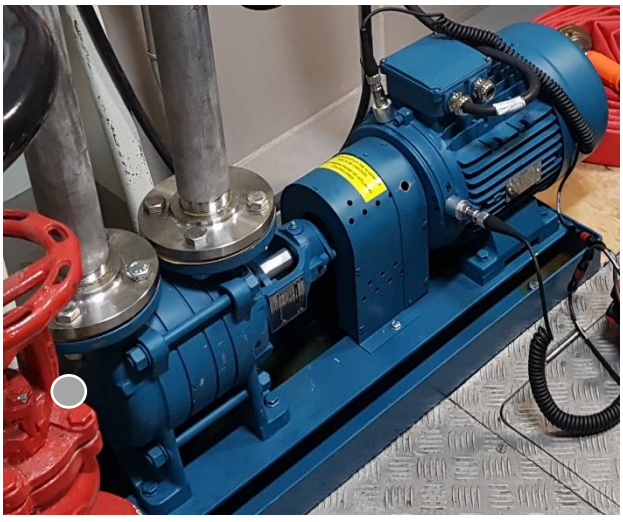


Figure 3-12: Loose Mounting of Test Pump

3.6.2.6. Unbalanced Shaft

A lathe was used to remove material from the shaft as shown in Figure 3-13 and Figure 3-14, 1mm from the top surface of the greyed area of the shaft to create a static imbalance. Approximately 1mm was removed and the machined surfaces polished. During this process, it was also noted that the shaft is slightly bent.

This modification to the shaft caused a severe leak in the pump as the packing ring could no longer seal. This meant that vibration and temperature data could not be collected from the pump–motor coupling position as the vibration probes were not waterproof. A guard was installed to manage the leak which obstructed temperature measurements. The issue and the guard are shown in Figure 3-15 and Figure 3-16. Vibration and temperature measurements from the pump casing were used as replacements. This fault has the potential to sink the vessel through the continuous leakage if left unchecked overnight.

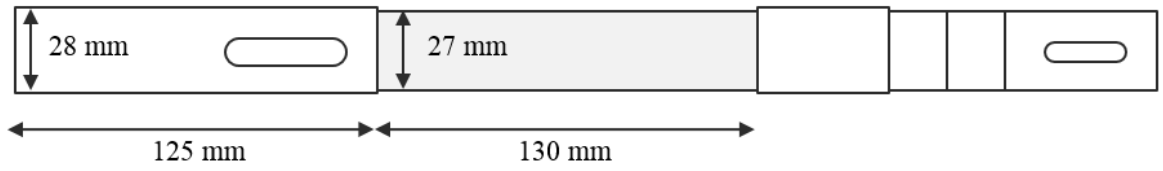


Figure 3-13: Unbalanced Shaft Diagram

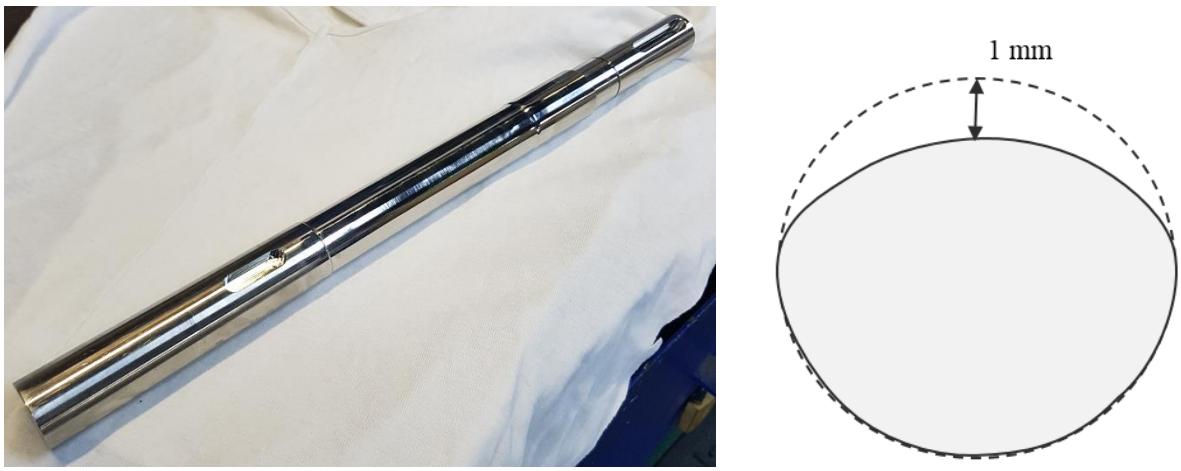


Figure 3-14: Left: Unbalanced Shaft Image, Right: Cross-section Diagram

Figure 3-15: Unbalanced Shaft Causing Leak in Test Pump

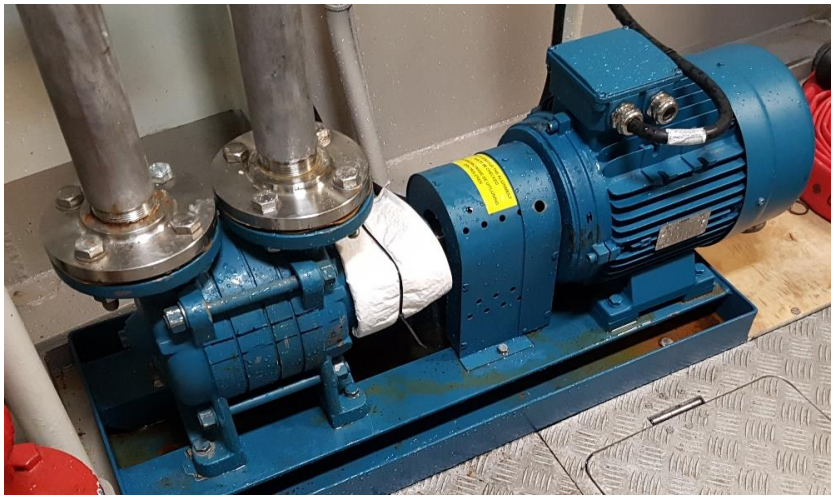


Figure 3-16: Temporary Guard Installed on Test Pump

3.6.2.7. Misaligned Shaft

The pump mounts were loosened, and the pump rotated anti-clockwise in the axial direction by approximately 1mm. Misalignment may also have occurred in the vertical direction due to this movement. This combination of misalignment in two directions is known more generally as an offset misalignment and is shown by the red line in Figure 3-17 as an approximate 1mm increase along the left-hand top edge of the coupling.

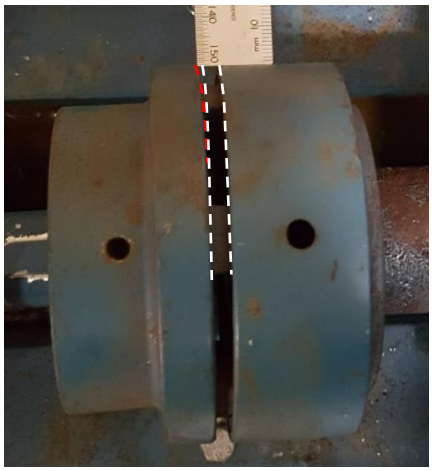


Figure 3-17: Offset Misalignment of Pump - Motor Coupling

3.6.3. CM Data Collection

The overall objective of the maintenance system was to minimize the risk of faults occurring in the Number 2 General Service Pump. Built using the experimental datasets, the system can determine the probabilities of the seven faults within the Number 2 General Service Pump if it analyses similar data, which is termed CM data. Only one decision context was considered due to time and resource limitations. This was operation of the Number 2 General Service Pump while 'the vessel was alongside and no equipment running'. A similar 0 bar suction pressure and 2.1 bar discharge pressure were used which produced a seawater flowrate of approximately $20m^3hr^{-1}$.

CM data were collected according to Table 3-6 and Figure 3-18 (described in Table 3-8 and Table 3-9) from the Number 2 General Service Pump for 10 minutes fortnightly over a period of approximately six months between 15/03/18 and 2/08/2018. This monitoring period was selected as it corresponded to two periodic PM intervals. Completion of the monitoring resulted in five measurements per dataset for 11 datasets, although low quality measurements were removed. Low quality measurements resulted from external interference in some cases. A dataset of 51 measurements remained. The datasets describing the Number 2 General Service Pump are described hereafter as CM data. The results of CM data processing and analysis are discussed in Chapter 5.

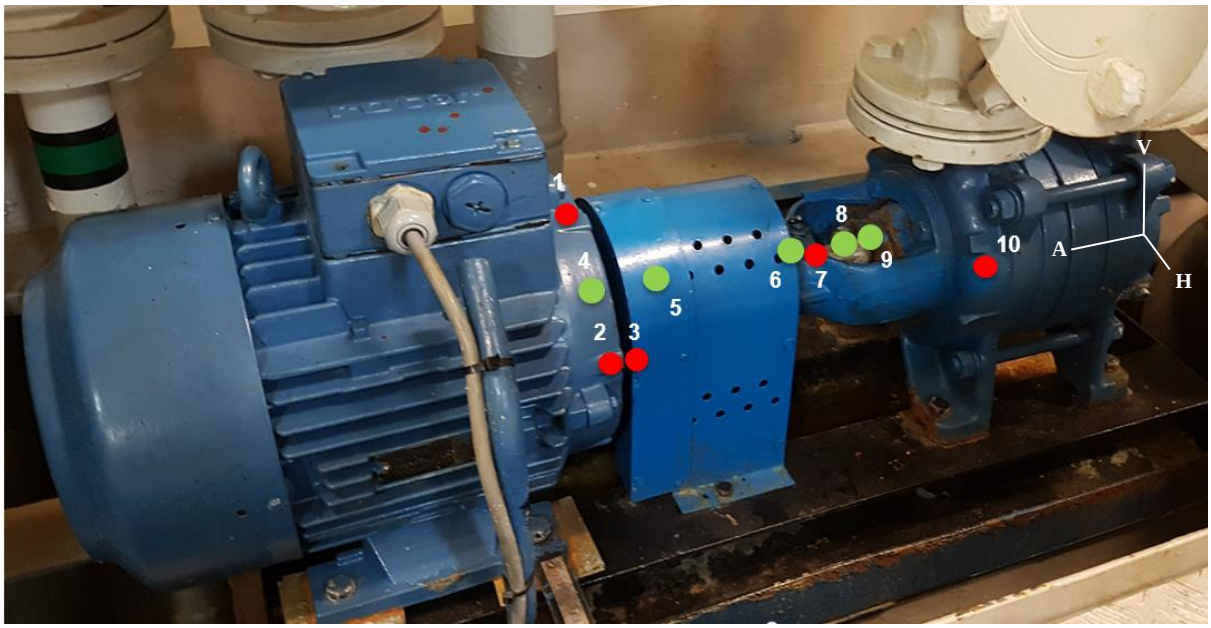


Figure 3-18: Number 2 General Service Pump Vibration and Temperature Measurement Points

Table 3-8: Description of Number 2 General Service Pump Vibration Measurement Points

POINT	**10**	**7**	**1**	**2**	**3**
LOCATION	Pump, Casing	Coupling Guard	Motor, Casing	Motor, Casing	Motor, Casing
ORIENTATION	Horizontal	Horizontal	Vertical	Horizontal	Axial

The 'Horizontal'(H), 'Vertical'(V) and 'Axial'(A) directions are indicated on Figure 3-18. These directions correspond to the orientations of the vibration measurement probes used to obtain each measurement.

Table 3-9: Description of Number 2 General Service Pump Temperature Measurement Points

POINT	**9**	**8**	**6**	**5**	**4**
LOCATION	Pump, Bearing Casing	Shaft	Coupling	Motor, Drive End Bearing	Motor, Casing

3.7. Stage 2b: *Data Processing*

This subsection provides a simplified description of how the ten experimental datasets comprising measurements in Table 3-6 were transformed into features used to create the discriminant classifier. The experimental data describe each of the fault states in Section 3.6.2, and can be used by each classifier to identify the probability of occurrence of each in unlabelled CM data. Classifiers are developed in Section 4.1. The same process as outlined in this section is applied to the CM data. The complete set of data processing steps are described in Appendix D.

Data processing was conducted according to the following five main steps:

1. *Extract and save vibration FFT data from the instrument for the locations indicated in* Table 3-4 *and velocity waveform data for Position 1 only.*
2. *Using the shaft RPM measurement taken for the current experiment or CM measurement, process all FFT files using a MATLAB script to obtain Excel files which contain the amplitudes of relevant peaks.*
3. *Process all waveform data through a MATLAB script to obtain Excel files which contain descriptive statistics of the waveform (Kamiel, 2015; Sakthivel et al., 2010).*
4. *Import thermal imaging data from thermal imaging camera into FLIR processing software, saving temperature data from each location as shown in* Table 3-5 *(or* Table 3-9 *for the Number 2 General Service Pump) per minute of image collection.*
5. *Combine the vibration, temperature and manually collected data into one dataset.*

The completion of the steps resulted in a 140 x 85 experimental dataset which described ten 'fault found' and 'no fault found' conditions and a 51 x 85 CM dataset.

3.7.1. Processing Vibration FFT Data

Conventional vibration analysis involves the extraction of features from velocity or acceleration FFT series data (Mobius Institute, 2015). Peak amplitudes at certain frequencies can be used to identify common centrifugal pump faults. Of interest to this study are the: Blade Pass Frequency (BPF), Ball Pass Frequency of the Outer Race (BPFO), and the fundamental frequency of

vibration f_0. Harmonics of vibration can be calculated as multiples of f_0, shown in Table 3-10. In the present work, velocity FFT peak values were measured in mms^{-1}. The following equations were used to evaluate the blade pass frequency, fundamental frequency and BPFO in the present study (Mobius Institute, 2015).

$$BPF = \frac{Shaft\ Rotation\ Speed \times Number\ of\ Impeller\ Blades}{60} \tag{3-1}$$

$$f_0 = \frac{Shaft\ Rotation\ Speed}{60} \tag{3-2}$$

$$BPFO \approx \left(\frac{Number\ of\ balls\ in\ bearing}{2} \right) - 1.2 \tag{3-3}$$

Table 3-10: Relations Between Vibration Harmonics and Fundamental Frequency

HARMONIC OF VIBRATION	RELATIONSHIP TO FUNDAMENTAL FREQUENCY
FIRST	$= f_0$
SECOND	$= 2 f_0$
THIRD	$= 3 f_0$
N^{TH}	$= n f_0$

The BPFO is a multiplication factor for fundamental frequency f_0. In (3-3) the number of balls in the bearing was 8 in both the test pump and air conditioning pump. It was required that BPFO was estimated as the contact angle of the balls in the bearing is unknown.

Upon analysis of the BPFO using the data from the damaged bearing experiment, a peak was evident which could be reasonably assumed to be the BPFO. However, the peaks shown at the second and third harmonics were much larger than those of the 'normal operation while the vessel alongside and no engines running' condition, and unique to the worn bearing condition (Refer to Appendix E, Figure E-6). Extraction of these harmonics was performed in lieu of using BPFO.

Using the shaft rotation speed obtained as part of the measurement scheme in Table 3-6 and pump data in Table 3-1, these characteristic vibration parameters were calculated by a MATLAB script applying equations (3-1) to (3-3).

Other characteristic parameters such as the Ball Pass Frequency Inner (BPFI) could also be used to characterise the faults. However, this study focused on the analysis of vibration harmonics as these are more easily generalised between the air conditioning and test pumps.

The experimental wear dataset showed characteristic peaks at the vibration harmonics (Refer to Appendix E) which provided further incentive to analyse these harmonics. The feature extraction process was automated, and the corresponding software is presented in Appendix C.

As there was some difference between the calculated values of the frequencies of peaks calculated using equations (3-1) to (3-3) and the measured values during the experiments, some inaccuracy in the vibration data was introduced due to either the Commtest vb7 or the tachometer. This difference is evident in Figures E-1 to E-10 in Appendix E. The difference in calculated and measured values have resulted in the feature extraction script overlooking important data.

Therefore, an error margin was included in each calculated value. These error margin values were determined using the deviation of the calculated values using equations (3-1) to (3-3) and Table 3-10 from the data collected in 'no fault' experiments (Refer to Appendix E, grey data in Figures E-1 to E-10). Values from the 'damaged bearing' experiment (Refer to Appendix E, Figure E-5) determined the error ranges in the case of the 12^{th}, 20^{th}, 36^{th} and 37^{th} order harmonics as these could not be determined using data from 'normal operation while vessel is alongside and no engines running' experiments as seen in the same figure. These slight differences are not evident in the plot, though are evident when looking at the data directly. Error margin values are shown in Table 3-11.

Table 3-11: Assumed Errors in Vibration Data

CHARACTERISTIC FREQUENCY PEAK	ASSUMED ERROR IN PEAK VALUES DUE TO INSTRUMENTS
BLADE PASS FREQUENCY (BPF)	± 0.5% or ~ ± 1 bin (1.25Hz)
FUNDAMENTAL FREQUENCY	± 1.5% or ~ ± 3 bins (4.5Hz)
SECOND HARMONIC	± 1.5% or ~ ± 3 bins (4.5Hz)
THIRD HARMONIC	± 1.5% or ~ ± 3 bins (4.5Hz)
FOURTH HARMONIC	± 1.5% or ~ ± 3 bins (4.5Hz)
FIFTH HARMONIC	± 1.5% or ~ ± 3 bins (4.5Hz)
SIXTH HARMONIC	± 1.5% or ~ ± 3 bins (4.5Hz)
SEVENTH HARMONIC	± 1.5% or ~ ± 3 bins (4.5Hz)
EIGHTH HARMONIC	± 1.5% or ~ ± 3 bins (4.5Hz)
12^{TH} HARMONIC	± 1.5% or ~ ± 3 bins (4.5Hz)
20^{TH} HARMONIC	± 1.5% or ~ ± 3 bins (4.5Hz)
36^{TH} HARMONIC (36.6 f_0)	± 1% or ~ ± 2 bins (3Hz)
37^{TH} HARMONIC (37.6 f_0)	± 1% or ~ ± 2 bins (3Hz)

13 features were obtained from each of the vibration FFT measurement locations on the relevant pump by the FFT feature extraction script (Refer to Table 3-4, Table 3-8 and Figure 3-11 and Appendix C), resulting in 65 features per sample. Values of zero replaced features which could not be calculated from the data due to measurement error or environmental interference.

Lastly, high frequency FFT or demodulated spectra are often used for fault detection in bearings (Mobius Institute, 2015). This study provides a general overview of vibration data for use with ML and does not incorporate this data. However, this data may improve the quality of the 'Damaged bearing' experimental dataset.

3.8. Summary

Chapter 3 described the application of a Number 2 General Service Pump aboard a naval tug, leading in to the development a pump RBM system. The Number 2 General Service Pump application and test pump configuration are were described in the present Chapter. The prior maintenance of the pump since 2015 was attached as Appendix B. The user prompt stage which determines the pump's environment which is important to the maintenance decision was then

described. Seven common centrifugal pump faults and three operational scenarios were simulated and measured using the test pump to produce labelled classification learning datasets. CM data were collected from the Number 2 General Service Pump for later classification using the system. As raw data were not suitable features for a supervised learning algorithm, significant features were extracted from the data. Appendices C – E provide the data processing software, processing steps and examples of raw vibration data.

The experimental data are then used to develop a set of linear discriminant classifiers in Chapter 4, followed by the development of a corresponding set of decision trees. The completed system can recommend actions given the pump CM data. The results of the CM data analysis are presented in Chapter 5.

CHAPTER 4 - Maintenance System Development Part II:

Stage 3 - Data Analysis, and Stage 4 - Storage and Transmission

The present Chapter 4 completes the description of how a maintenance system was created within the RBM framework outlined in Chapters 1 and 2 previously for the shipboard pump application. Prior description of the shipboard pump application, four-stage system workflow and its first two stages; User Prompt, Data Collection and Processing was provided in Chapter 3. The remaining two stages in this process in are described the present Chapter 4. These include Data Analysis using linear discriminant classification and decision trees, and considerations which should be given to data in the fourth Storage and Transmission stage. Much of the present Chapter describes the Data Analysis stage which was conducted using MATLAB software. The system software is attached alongside the Data Processing software Appendix C. Lastly, calculation procedures for the metrics used to measure the performance of system against periodic PM are described. The results of the performance measurement and are presented and discussed in Chapter 5.

4.1. Stage 3: *Data Analysis*

4.1.1. Developing Linear Discriminant Analysis Classifiers

The RBM framework developed for marine vessels in Chapter 1 specifies that a probabilistic model is required as part of the Risk Assessment element. As it has been identified an effective general technique to develop such a model, Multivariate Analysis (MVA) and Linear Discriminant Analysis (LDA) as a supervised classification method were described in Chapter 2 previously.

The classification algorithm was implemented in MATLAB 2018a for the shipboard pump application outlined previously in Chapter 3. The structure of the classification model was described previously in Chapter 2. The classes in the model corresponded to the common causes of failure identified previously for centrifugal pumps, in addition to the operational modes of the vessel. The experimental datasets obtained as learning data for the classifiers are summarised Table 4-1. Datasets 1-3 represent 'no fault found' conditions, while datasets 4-10 represent 'fault found conditions'.

Table 4-1: Summary of Experimental Datasets

DATASET	FAULT CONDITION	SIZE
1	Vessel Alongside, No Engines Running	85 Features x 50 Samples
2	Vessel Alongside, Engines Running	85 Features x 10 Samples
3	Vessel at Sea	85 Features x 10 Samples
4	Worn Impeller	85 Features x 10 Samples
5	Loose Packing	85 Features x 10 Samples
6	Damaged Pump Drive-End Bearing	85 Features x 10 Samples
7	Worn Pump Drive-End Bearing	85 Features x 10 Samples
8	Loose pump foot	85 Features x 10 Samples
9	Static Imbalance in Shaft	85 Features x 10 Samples
10	Offset Misalignment in Shaft	85 Features x 10 Samples

Four classifiers were developed using all datasets described in Table 4-1. The four distinct classifiers represented four operational scenarios as follows:

- C1 – Pump operating with no fault, vessel alongside, engines quiet
- C2 – Pump operating with no fault, vessel alongside, engines running
- C3 – Pump operating with no fault, vessel slow steaming (<4 knots) in the harbour (at sea), non-emergency case
- C4 – Pump operating with no fault, vessel slow steaming (<4 knots) at sea, emergency case.

C3 and C4 were created using the same learning data and produce the same probability results, suggesting that C4 appears to be redundant. However, C4 has been created to simplify the system design as there are four contexts and four decision trees. A fourth classifier is also useful in future system development if the learning data must be changed or updated in this context.

The datasets used from Table 4-1 and classes within each of the classifiers are detailed in Table 4-2 to Table 4-5.

Table 4-2: Development of Linear Discriminant Classifier C1

CLASSIFIER C1		
DATASET(S)	**CLASS NUMBER c_k**	**CLASS DESCRIPTION**
1	1	Pump operating with no fault, vessel alongside, engines quiet
4	2	Worn Impeller
5	3	Loose Packing
6	4	Damaged Pump Drive-End Bearing
7	5	Worn Pump Drive-End Bearing
8	6	Loose pump foot
9	7	Static Imbalance in Shaft
10	8	Offset Misalignment in Shaft

Table 4-3: Development of Linear Discriminant Classifier C2

CLASSIFIER C2		
DATASET(S)	**CLASS NUMBER c_k**	**CLASS DESCRIPTION**
1,2,3	1	Pump operating with no fault, vessel alongside, engines running
4	2	Worn Impeller
5	3	Loose Packing
6	4	Damaged Pump Drive-End Bearing
7	5	Worn Pump Drive-End Bearing
8	6	Loose pump foot
9	7	Static Imbalance in Shaft
10	8	Offset Misalignment in Shaft

Table 4-4: Development of Linear Discriminant Classifier C3

CLASSIFIER C3		
DATASET(S)	**CLASS NUMBER c_k**	**CLASS DESCRIPTION**
1,2,3	1	Pump operating with no fault, vessel slow steaming (<4 knots) in the harbour (at sea), non-emergency case
4	2	Worn Impeller
5	3	Loose Packing
6	4	Damaged Pump Drive-End Bearing
7	5	Worn Pump Drive-End Bearing
8	6	Loose pump foot
9	7	Static Imbalance in Shaft
10	8	Offset Misalignment in Shaft

Table 4-5: Development of Linear Discriminant Classifier C4

CLASSIFIER C4		
DATASET(S)	**CLASS NUMBER c_k**	**CLASS DESCRIPTION**
1,2,3	1	Pump operating with no fault, vessel slow steaming (<4 knots) in the harbour (at sea), emergency case
4	2	Worn Impeller
5	3	Loose Packing
6	4	Damaged Pump Drive-End Bearing
7	5	Worn Pump Drive-End Bearing
8	6	Loose pump foot
9	7	Static Imbalance in Shaft
10	8	Offset Misalignment in Shaft

Datasets 1, 2 and 3 were merged to create a larger dataset for Class 1 for C2, C3 and C4. Otherwise singular covariance matrices would result due to insufficient data and highly-correlated features. Weighting coefficients (Refer to Chapter 2) according to Table 4-6 were used to merge datasets which did not correspond to Class 1 in each classifier. Weighting coefficients for all remaining classes were set equal to 1. C3 and C4 were assigned identical weighting coefficients for the reasons mentioned previously.

Table 4-6: Weighting Coefficients for Class 1

WC_1	DATASET	SIZE	ELEMENT VALUES	SIZE WC_1
C2	1	50	0.6	70
	2	10	1	
	3	10	0.2	
C3/C4	1	50	0.1	70
	2	10	0.2	70
	3	10	1	70

4.1.1.1. Resubstitution Performance

The resubstitution performance of these classifiers was estimated using confusion, certainty and doubt matrices. Custom functions were written to compute confusion, certainty and doubt matrices in MATLAB following the calculations described previously in Chapter 2. All of these are 8 x 8 matrices, with each row and column corresponding to one class in Table 4-2.

Four identical confusion matrices were produced using MATLAB. All of the Main Diagonal (MD) elements of each matrix were evaluated as 100, while all Off Diagonal (OD) elements were evaluated as 0. These results indicate that each of the classifiers determine the correct class of the training data sample 100% of the time.

Additionally, four identical certainty matrices were produced. All of the MD elements of each matrix were evaluated as 100 and all OD elements evaluated as NaN. These results indicate that each of the classifiers are 100% confident or certain that every label has been correctly assigned.

Lastly, four approximately identical doubt matrices were produced. All of the MD elements of each matrix were evaluated as zero or approximately zero. The largest value was 2.03E-15. Similarly to the certainty matrices, all OD elements were evaluated as NaN. In alignment with the certainty matrix results, these doubt matrix results indicate that the classifiers have approximately zero diffidence or doubt that they have assigned the correct class labels to the training data samples.

The results of the resubstitution performance tests suggest that the classifiers possess excellent resubstitution or learning sample classification performance, however 'hold-out' performance of these should also be estimated as the classifiers will be used to estimate probabilities based on data which were not trained.

4.1.1.2. Hold-out Performance

The 'hold-out' performance of the classifiers was estimated using increasingly noisy data based on the learning samples. The noisy data is intended to simulate the CM data obtained from the Number 2 General Service Pump. Two calculations were necessary to create noisy data. Firstly, the mean distance between the means of all 85 features in each vector of learning data was calculated. Secondly, each mean vector and a specified variance were used as parameters to randomly generate a new noisy vector. The noisy vector is then classified to produce probabilities. This analysis does not strictly determine the hold-out performance of the classifiers but a similar measure as the new noisy vectors are altered versions of the learning data, though are very different to the learning data when generated with a high variance.

The previous resubstitution performance of the classifiers is the same as the 'hold-out' performance on data with 0% noise for comparison. Performance estimation of each classifier was performed using the confusion, certainty and doubt matrices created from the classification of the noisy vectors for each classifier. The average values in Table 4-7 are used to summarise the performance estimation results. Average values are reported for the MD of the confusion matrix, the MD of the certainty and doubt matrices and the OD of the certainty and doubt matrices. Noise

levels of 10%, 50% and 100% were used. The individual matrices are provided in Appendix F. Again, principles behind each of the matrices were discussed in Chapter 2 previously, and classifiers C3 and C4 are equivalent. An example calculation follows the table.

Table 4-7: 'Hold-out' Classification Performance using Noisy Data

CLASSIFIER	NOISE LEVEL	CONFUSION	CERTAINTY		DOUBT	
		MD	MD	OD	MD	OD
C1	10%	100.00	100.00	0	1.13E-20	0
	50%	78.50	100.00	99.78	7.84E-04	1.91E-01
	100%	58.50	100.00	99.80	2.73E-09	5.18E-02
C2	10%	100.00	100.00	0	2.57E-11	0
	50%	73.75	99.10	99.90	9.02E-01	1.02E-01
	100%	47.50	100.00	99.47	2.65E-03	4.41E-01
C3	10%	100.00	100.00	0.00	6.35E-12	0
	50%	63.75	100.00	100.00	1.54E-03	5.18E-12
	100%	36.25	75.00	100.00	1.82E-36	3.71E-17
C4	10%	100.00	100.00	0	6.35E-12	0
	50%	63.75	100.00	100.00	1.54E-03	5.18E-12
	100%	36.25	75.00	100.00	1.82E-36	3.71E-17

Example Calculation:

The following general example can be applied to the relevant matrix to calculate all values in Table 4-7. The highlighted values in Table 4-7 can be calculated as the average of the MD and OD from doubt matrix in Figure 4-1 (Reproduced from Appendix F). The matrix represents the doubt of C3 given vectors with a 50% Gaussian Noise (GN) level.

Figure 4-1: C3 Doubt Matrix, 50% GN

6.54E-26	1.04E-21	NaN	9.36E-117	1.00E-50	0	NaN	NaN
6.20E-110	2.03E-46	NaN	0	NaN	NaN	5.44E-23	NaN
NaN	0	6.40E-11	0	1.34E-33	NaN	NaN	NaN
NaN	8.21E-179	NaN	6.64E-06	NaN	NaN	NaN	NaN
NaN	NaN	NaN	NaN	5.05E-266	NaN	NaN	NaN
8.81E-11	NaN	1.47E-41	1.47E-90	NaN	1.23E-02	NaN	NaN
NaN	0	NaN	2.64E-123	NaN	NaN	1.17E-35	NaN
NaN	2.30E-81	NaN	NaN	NaN	NaN	NaN	7.27E-57

The MD value is the average of the dark grey elements, while the OD value is the average of the light grey elements. ‘NaN’ elements are excluded in these calculations.

Table 4-7 shows that the MD confusion decreases while the OD certainty and doubt of all classifiers increase as the noise level in the input data increases. This indicates a reduction in classifier performance as the classifiers become more certain about making incorrect classifications. The MD certainty of C1 and C2 is still quite high under the 100% noise condition although there is a corresponding increase in OD certainty as the classifiers are misclassifying 40% or more of the samples. C3 and C4 are performing as expected, having misclassified all samples from Class 6 which leaves the certainty and doubt matrices undefined. As discussed previously, C3 and C4 are equivalent.

4.1.2. Decision Analysis of Maintenance of the Number 2 General Service Pump

Chapter 2 previously discussed how Decision Theory and MAUT may be used to make rational decisions under risk and that decision trees with lotteries at their nodes may be used to capture this risk. Then, a solution was developed to find the expected utility for the special case where a lottery is a recursive ICOL. Decision trees containing ICOLs are developed in the present section for the Number 2 General Service Pump application. The decision trees are then combined with the linear discriminant classifiers to form the complete maintenance system.

The parameters of the decision analysis are defined in the remainder of the present Section 4.1.2. The methods which were used to obtain expert data and develop the MAU function in are defined in Section 4.1.3 and lastly the lottery models within the decision trees and their expected utility solutions in are defined in Section 4.1.4. The completed system can then be used to create a maintenance schedule using the CM data. This schedule is generated and compared to the corresponding periodic PM schedule using availability and overall maintenance cost in Chapter 5.

4.1.2.1. Decision Contexts

The expert involved in the decision analysis for the pump was the Chief Engineer of the vessel as he was directly responsible for performing the maintenance of the pump and has over 10 years of marine engineering expertise.

Consultation with the expert enabled the identification of 16 unique decision contexts in relation to maintenance of the pump. The contexts are the combinations of the following:

- ***Location of the ship:*** At Sea or Alongside
- ***The current situation facing the ship:*** Emergency or Not Emergency
- ***Duration of the job required of the No. 2 General Service Pump:*** Less than 6 hours or Greater than 6 hours
- ***Function required of the No. 2 General Service Pump:*** Delivery of fluid or Suction

The present work considers only vessel location and simulate an emergency context due to limitations on the collection of further data. The remaining 12 contexts provide avenues for future work using the present system.

Four decision trees were developed which described the following four contexts, following on from those used to develop the linear discriminant classifiers previously. 'Emergency' in this instance refers to the inability of the crew to stop the pump for maintenance.

- ***Context 1:*** Vessel Alongside, Pump Running Alone in Engine Room
- ***Context 2:*** Vessel Alongside, Pump Running and Engines Running in Engine Room
- ***Context 3:*** Vessel Slow-Steaming (<4 knots) in the harbour (at sea), Pump Running, Not an Emergency
- ***Context 4:*** Vessel at Sea, Pump, Emergency

4.1.2.2. Maintenance Actions and Decision Trees

Given the specific context, the general structure of each decision tree for the maintenance of the pump consists of the decision point at its root, branching out directly to any number of

maintenance actions. The outcomes of the maintenance actions at the nodes are uncertain and can be modelled as lotteries (Clemen, 1996).

There are five possible maintenance actions for the pump in the first three contexts. These are listed in Table 4-8. The action A_1 is different when considering the location of the ship. A_5 in Table 4-8 is labelled 'Do nothing' as the Number 2 General Service Pump is routinely inspected by the expert every 24 hours. The fourth context is detailed in Table 4-10 and contains only the deferral actions as the pump must be operational in an emergency to ensure personnel and vessel safety.

Table 4-8: Maintenance Actions for Number 2 General Service Pump, Contexts 1-3

NUMBER	**ACTION**	
	Context 1-2	***Context 3***
A_1	Stop No.2 Pump to check inside	Stop No. 2 Pump to check inside, change to No. 1 Pump immediately
A_2	Inspect pump more than once in 6 hours	Inspect pump more than once in 6 hours
A_3	Inspect pump after 6 hours	Inspect pump after 6 hours
A_4	Inspect pump after 12 hours	Inspect pump after 12 hours
A_5	Inspect pump after 24 hours / Do nothing	Inspect pump after 24 hours / Do nothing

Table 4-9: Maintenance Actions for Number 2 General Service Pump, Context 4

NUMBER	**ACTION**
	Context 4
A_2	Inspect pump more than once in 6 hours
A_3	Inspect pump after 6 hours
A_4	Inspect pump after 12 hours
A_5	Inspect pump after 24 hours / Do nothing

The four decision trees are shown in Figure 4-2. The superscript values correspond to the decision context and highlight that the calculations within the lotteries change with the decision context. The expert's subjective estimates change between contexts, and these are also used in the

calculation of utility values. Also, the decision trees for Contexts 1-3 illustrate that one of five Policies may be calculated as the action possessing the maximum expected utility, while the decision tree for Context 4 shows that one of four Policies may be selected from the four possible actions. Policies in the remaining discussions in this work represent the number of the action possessing the maximum expected utility, or alternatively the number of the action which is most rationally performed in the given decision context.

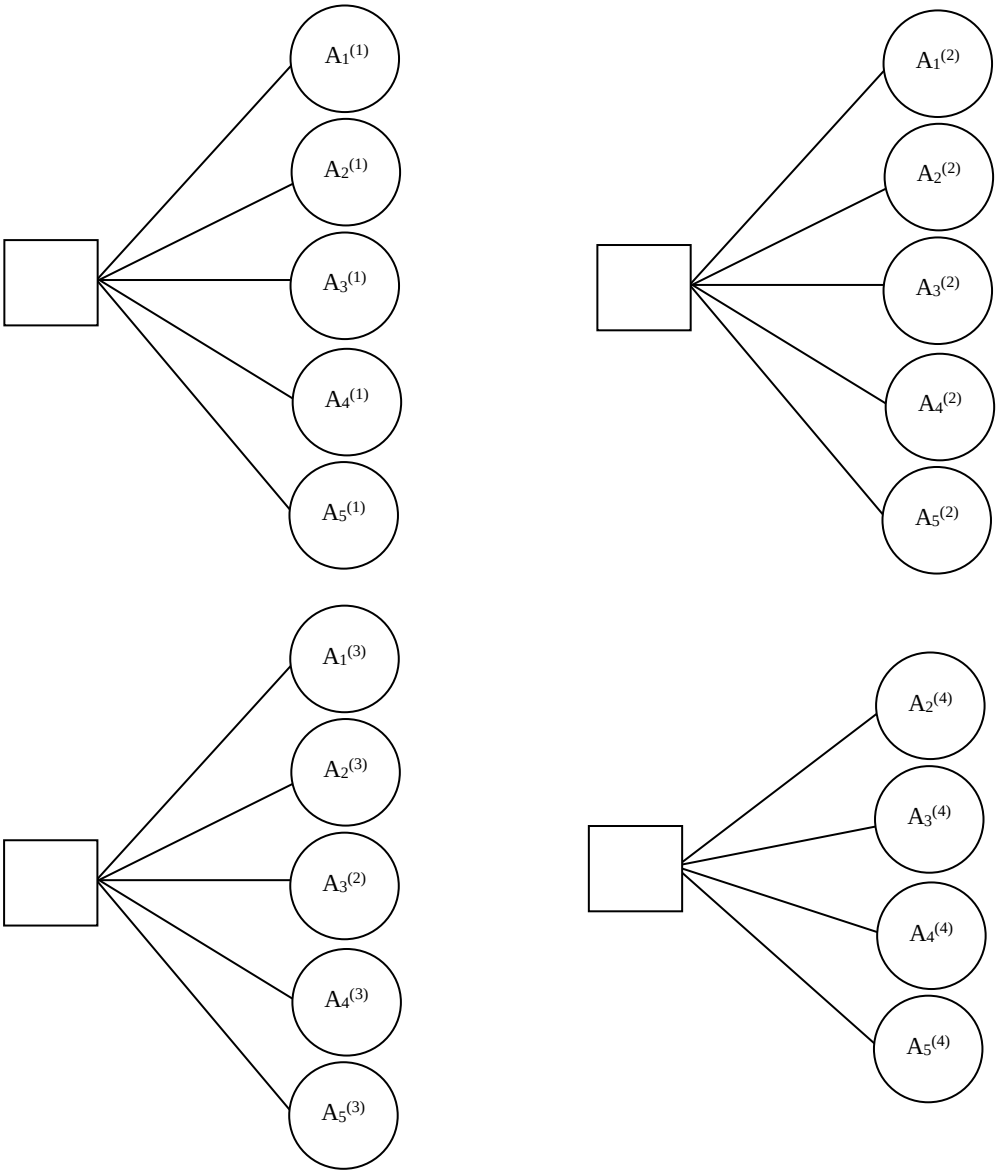


Figure 4-2: Decision Trees. Clockwise from top: Context 1, Context 2, Context 4 and Context 3

To find expected utility solutions for the lottery models, a measure of the probability of different conditions occurring in the pump is required, alongside the overall utility of each consequence which corresponds with the expert's preferences. Obtaining probability information using linear discriminant classifiers was discussed previously in Section 4.1.1.

Classifier C1 corresponds to Context 1, C2 for Context 2, and lastly C3 and C4 correspond to Contexts 3 and 4 respectively. Each class of these probabilities translates into a state of nature in Table 4-10. The relevant classifier calculates $P(\theta_j \mid \vec{x})$ given some measurement $\vec{x}$. The experts' preferences are captured and developed into the multi-attribute utility of each consequence in Section 4.1.3.

Table 4-10: States of Nature for Number 2 General Service Pump

CLASS NUMBER c_k	**STATE OF NATURE** $P(\theta_j \mid \vec{x})$	**DESCRIPTION**
1	$P(\theta_1 \mid \vec{x})$	No Fault Found
2	$P(\theta_2 \mid \vec{x})$	Worn Impeller
3	$P(\theta_3 \mid \vec{x})$	Loose Packing
4	$P(\theta_4 \mid \vec{x})$	Damaged Pump Drive-End Bearing
5	$P(\theta_5 \mid \vec{x})$	Worn Pump Drive-End Bearing
6	$P(\theta_6 \mid \vec{x})$	Loose Pump Foot
7	$P(\theta_7 \mid \vec{x})$	Static Imbalance in Shaft
8	$P(\theta_8 \mid \vec{x})$	Offset Misalignment in Shaft

On occasion, the situation where one or more $P(\theta_j \mid \vec{x}) = 0$ can occur, which prevents further calculations. A slight correction to the value would enable calculations to continue. An approach to the problem is presented as part of the software in Appendix C.

4.1.2.3. Consequences

The development of the lotteries followed the development of the four decision trees. Within the lotteries each action is linked to its outcomes. Discussion with the expert enabled the creation of a qualitative description of each consequence resulting from the four or five maintenance actions in the four decision trees. These are tabulated in Appendix G.

Each consequence was then redefined as a vector of attributes $\vec{y}$. Each $\vec{y}$ was comprised of six measurable attributes y_a where $a = 1,2,3...6$: Downtime, Expected repair cost of the pump, Expected repair cost of the vessel, Expected number of people severely injured, Routine Maintenance Cost, and Lloyds Compliance. In this case, 'severe injury' was defined as a 4.5 bar spray of seawater experienced by personnel causing a fall and hospitalisation resulting in time off work due to the fall.

Some reasonable assumptions were required to calculate each y_a:

- All resources are available at the time of the maintenance decision.
- Maintenance results in complete repair and does not introduce further failures.
- The pump is required to deliver seawater for a maximum duration of six hours.
- Loss of life due to failure of the pump is not possible.
- The vessel has a maximum crew of four.

The six attributes are described further in Table 4-11.

Table 4-11: Attributes of Consequences

ATTRIBUTE	y_1 - DOWNTIME	y_2 – EXPECTED REPAIR COST OF PUMP	y_3 - EXPECTED REPAIR COST OF SHIP	y_4 – EXPECTED NUMBER OF PEOPLE SEVERELY INJURED	y_5 - ROUTINE MAINTENANCE COST	y_6 - LLOYDS COMPLIANCE
DESCRIPTION	The time that the pump is available to complete the current task, due to maintenance or failures.	The expected cost to repair the pump, represented as a fraction of the replacement cost of a new identical pump-motor set. This is an additional cost beyond routine maintenance of the pump.	The expected cost to repair the ship, represented as a fraction of the replacement cost of a new identical vessel. This is an additional cost beyond routine maintenance of the pump.	The expected probability that up to four people are severely injured due to pump failure.	The complete cost of maintenance of the pump for the given state of nature to restore function and complete the current task. This includes time, parts, tools and labour to perform a maintenance task. The cost of the time involved is calculated from Downtime.	Binary measure of current vessel compliance with Lloyds insurance conditions. 2 of 3 of main fire pump, No.1 and No.2 pumps must be operational. Vessel cannot leave the wharf if found non-compliant due to failure of No.2 pump resulting in contract breach.
UNIT	Time, hours	$AUD	$AUD	Number of People	$AUD	No units
VALUE RANGE	0 - 4.5 hours	0 - 4253.08	0 - 10,000,000	0 - 4	0 - 4253.08	0 or 1
COMMENTS	Downtime is only the time elapsed during the repair and recommissioning of No.2 if decision context does not warrant changeover to No. 1.	Replacement cost of pump and motor is 4221.37 $AUD.	Replacement of the whole vessel is $10 million AUD.			

4.1.3. Obtaining Expert Data: *Multi-Attribute Utility and Subjective Probability*

Past trends in maintenance research (Refer to Chapter 1) have highlighted the value of expertise in ordering tasks to create a maintenance schedule. New maintenance methodologies should continue to incorporate expertise. Decision Theory is applied in the present work to achieve this. Decision Theory (Clemen, 1996; Keeney & Raiffa, 1993) provides a structured way of surveying an expert to quantify his expertise, specifically by determining his preferences toward each consequence of his decision. Preferences are measured with utility.

Each consequence is defined as a multi-attribute vector $\vec{y}$, and requires a corresponding multi-attribute utility over each consequence to determine complete the calculations for each lottery and decision tree. The expert's preference information can be used to create a Multi-Attribute Utility (MAU) function over all attributes in Table 4-11.

The MAU function is a composite of the scaling constant k_a of each attribute y_a, and the expert's attribute utility functions over each y_a. Attribute utility functions describe the utility of different levels within each continuous attribute y_a such as cost in $ AUD or time in hours. Otherwise, utility values may be directly assigned to different levels of discrete attributes. y_1 to y_5 are continuous attributes as they are described by a continuous range of values whereas y_6 is not as it is described by a binary set of values. Values of 1 and 0 in y_6 are the best and worst outcomes of this attribute and were directly assigned utility values of 1 and 0 respectively.

The construction of attribute utility functions required the elicitation of consequence value intervals between y_{down} and y_{up} for given values of an attribute utility $u(y_a)$. Using the Certainty Equivalence method (Chapter 2) within the Triple Bisection Method (Tenekedjiev, 2007a) to shorten the survey time, consequence value intervals for $u(y_a)$ where $u(y_a)$ is assigned as 0, 0.2, 0.4, 0.5, 0.6 and 0.8, for all attributes $a = 1, 2, 3...6$ were obtained from the expert. The elicitation process is described in French (1986). The maximum and minimum values of the attributes were

obtained from Table 4-11. y_{down} and y_{up} were determined to the precision specified in Table 4-12. The Certainty Equivalence method was used for y_1, y_2, y_3 and y_5, while the Probability Equivalence method was used for y_4.

Table 4-12: Decision Precision Attribute Values

ATTRIBUTE	PRECISION
y_1 - Downtime	0.45 hrs
y_2 – Expected Replacement Cost of Component	$425 AUD
y_3 – Expected Replacement Cost of Vessel	$1,000,000 AUD
y_4 – Expected number of People Severely Injured	1 person
y_5 - Routine Maintenance Cost	$425 AUD

Several subjective probabilities were also required to solve the lotteries and ICOLs. These are described in Section 4.1.4. Prior discussion in Chapter 2 highlighted that it is necessary to use a numerical approach and a reference experiment to elicit subjective probabilities (Haase et al., 2013). The expert was surveyed using an 'urn of balls' reference experiment (Nikolova et al., 2004) which can determine subjective probabilities with an accuracy of ±1%. The resulting subjective probabilities are tabulated alongside the relevant lottery in Section 4.1.4.

At the completion of the utility elicitation, it was found that the experts' preferences are strictly decreasing for y_1 to y_5. Each attribute utility function $u(y_a)$ was then approximated as an arctan function for strictly decreasing preferences (Nikolova et al., 2018), which is described in (4-1) below. Parameters for the arctan utility functions were calculated using the Weighted Least Squares method. Parameters are tabulated in Table 4-13.

$$u(y_a) = 1 - \frac{\arctan(Ay - AB) - \arctan(Ay_{\min} - AB)}{\arctan(Ay_{\max} - AB) - \arctan(Ay_{\min} - AB)} \tag{4-1}$$

Table 4-13: Attribute Utility Function Parameters

ATTRIBUTE	FUNCTION	PARAMETERS	FITTING ERROR
y_1	Arctan	A = 0.5324 B = 2.2680 y_{max} = 4.5 y_{min} = 0	0.1344
y_2	Arctan	A = 5.5608×10^{-4} B = 2.0451×10^{3} y_{max} = 4.2531×10^{3} y_{min} = 0	0.1589
y_3	Arctan	A = 1.9113×10^{-9} B = 4.0839×10^{4} y_{max} = 10×10^{6} y_{min} = 0	1.8989×10^{-7}
y_4	Arctan	A = 28.3673 B = -8.0896 y_{max} = 4 y_{min} = 0	0.8486
y_5	Arctan	A = 5.7245×10^{-4} B = 2.1530×10^{3} y_{max} = 4.2531×10^{3} y_{min} = 0	0.1408
y_6	No model	Not applicable	N/A

The expert was indifferent to the values initialised for the attribute function y_3 during the analysis and the certainty equivalent within the specified decision precision in Table 4-12. This resulted in a linear utility function. The application of the Probability Equivalence resulted in a curved function for y_4.

Each y_a within each consequence to could now be measured and used to determine $u(y_a)$ using the attribute functions. The measurement of y_a depended on the decision context and the action resulting in that consequence. In some cases, y_a varied based on the probabilities $P(\theta_j \mid \vec{x})$ of

each of the states occurring in Table 4-10 as they are expected values. The translation of the qualitative descriptions of the consequences in Appendix G into y_a is described in Appendix H.

To obtain the MAU for each consequence, a MAU function was constructed assuming some combination of the six attribute utility functions, which is either additive or multiplicative according to the sum of the scaling constants k_a (Nikolova et al., 2008a; Nikolova et al., 2008b). An additive combination results when the sum of these values is equal to one, otherwise a multiplicative function is used. The additive and multiplicative types of MAU function are shown in (4-2) and (4-3) respectively.

$$U\left(y_{1,}y_2...y_a\right)=\sum_{1}^{a}Kk_a u\left(y_a\right) \qquad \text{if} \qquad \sum_{a=1}^{6}k_a=1 \qquad (4\text{-}2)$$

$$U\left(y_{1,}y_2...y_a\right)=\frac{\prod_{1}^{a}\left(K\,k_a\,u\left(y_a\right)+1\right)-1}{K} \qquad \text{if} \qquad \sum_{a=1}^{6}k_a\neq 1 \qquad (4\text{-}3)$$

In (4-2) and (4-3) : K is the overall scaling constant; $U\left(y_{1,}y_2...y_a\right)$ or $U\left(\vec{y}\right)$ is the MAU for a vector of a attributes y_1 to y_a. Lastly, k_a and $u\left(y_a\right)$ are the scaling constant and utility function of attribute y_a.

The scaling constants were elicited using the Probability Equivalence method and the corner consequences. However, these values showed such a significant bias (utility range = 0.99 to 1) toward the maximization of y_4. This meant that that the values of all other aspects were negligible (utility ranges 0.01 to 0).

Therefore, substitute k_a values were used according to Table 4-14. These values balance the aspects of the consequences, without changing the preference order elicited from the expert.

Table 4-14: Corner Consequences and Substitute k_a Values

ATTRIBUTE	MEANING	$u(y_a) = k_{a,\,lower}$	$u(y_a) = k_{a,\,upper}$
$y_{1,Corner}$	Downtime	0.11	0.15
$y_{2,Corner}$	Expected Repair Cost due to Component Failure	0.14	0.18
$y_{3,Corner}$	Expected Repair Cost due to Ship Sinking	0.18	0.22
$y_{4,Corner}$	Expected Number of People Severely Injured	0.68	0.72
$y_{5,Corner}$	Routine Maintenance Cost	0.07	0.11
$y_{6,Corner}$	Lloyds Restriction on Ship Operation	0.16	0.2

Solutions for all $k_{a,\,est}$ and K were found using an analytical solution of the non-uniform method in MATLAB (Nikolova, 2012). It was found that $k_{a,\,est} = 0.13, 0.16, 0.2, 0.7, 0.09, 0.18$ and $\sum_{a=1}^{6} k_a \neq 1$ with $p = 0$, meaning that the MAU function should be created using the multiplicative function form. The analytical non-uniform method was then applied to determine $K = -0.7276$. The overall MAU function could then be written as (4-4).

$$U\left(y_1, y_2 \ldots y_6\right) = \frac{\prod_{a=1}^{6}\left(-0.7276\, k_a\, u\left(y_a\right) + 1\right) - 1}{-0.7276} \tag{4-4}$$

Each $u\left(y_a\right)$ and $k_{a,est}$ were then used to produce $U\left(y_1, y_2 \ldots y_6\right)$ or $U\left(\vec{y}\right)$ for each consequence, placed on a scale of 0 to 1. A function was created to automatically calculate MAU values $U\left(\vec{y}\right)$ from attribute weights y_a.

Given the decision context and $P\left(\theta_j \mid \vec{x}\right)$ generated by the discriminant classifiers in Section 4.1.1, the appropriate decision tree is then selected and evaluated according to the maximum expected utility rule to produce a Policy recommendation. The lottery models representing each action and their methods of evaluation within each tree are presented in the next section.

4.1.4. Expected Utility from Lottery Models

Lottery models are developed in the present section for each of the actions shown as part of the decision trees previously in Figure 4-2. In some cases, these lottery models are ICOLs which can be simplified and their expected utility solutions derived (in the recursive case) according to the methods presented previously in Section 2.2.5.1. Otherwise, the expected utility solutions of FCOLs may be derived using current theory.

4.1.4.1. Lottery Model for A_1, Contexts 1 and 2

Action A_1 when the vessel is alongside (Refer to Table 4-8) was modelled as an FCOL which contains additional simple lotteries $l_{1\alpha}$ where $\alpha = 1...8$ as shown in Figure 4-3. Simple lotteries form the consequences of the FCOL as the expert prefers a different consequence based on the repair time of the equipment, specifically whether it is greater or less than 30 minutes. Conditional likelihoods of these repair times are captured in the simple lotteries as $P\left(R_{>30} \mid \theta_j\right)$ and $P\left(R_{<30} \mid \theta_j\right)$, with $R_{>30}$ and $R_{<30}$ representing repair times of greater than and less than 30 minutes respectively. The experts' preference toward each consequence is obtained by evaluating the MAU of each consequence using the function developed in Section 4.1.3.

$$L(A_1) = \left\langle \begin{matrix} P(\theta_1 \mid \vec{x}),\ l_{11}\ ;\ P(\theta_2 \mid \vec{x}),\ l_{12}\ ;\ P(\theta_3 \mid \vec{x}),\ l_{13}\ ;\ P(\theta_4 \mid x),\ l_{14}\ ;\ ... \\ P(\theta_5 \mid \vec{x}),\ l_{15}\ ;\ P(\theta_6 \mid \vec{x}),\ l_{16}\ ;\ P(\theta_7 \mid \vec{x}),\ l_{17}\ ;\ P(\theta_8 \mid x),\ l_{18} \end{matrix} \right\rangle$$

Figure 4-3: Compound Lottery $L\left(A_1\right)$, Contexts 1 and 2

The simple lotteries l_{11} to l_{18} are defined as follows:

$$l_{11} = \left\langle P\left(R_{>30} \mid \theta_1\right), U\left(\vec{y}_1\right);\ P\left(R_{<30} \mid \theta_1\right), U\left(y_2\right) \right\rangle,$$

$$l_{12} = \left\langle P\left(R_{>30} \mid \theta_2\right), U\left(\vec{y}_3\right);\ P\left(R_{<30} \mid \theta_2\right), U\left(y_4\right) \right\rangle,$$

$$l_{13} = \left\langle P\left(R_{>30} \mid \theta_3\right), U\left(\vec{y}_5\right);\ P\left(R_{<30} \mid \theta_3\right), U\left(y_6\right) \right\rangle,$$

$$l_{14} = \left\langle P\left(R_{>30} \mid \theta_4\right), U\left(\vec{y}_7\right);\ P\left(R_{<30} \mid \theta_4\right), U\left(y_8\right) \right\rangle,$$

$$l_{15} = \left\langle P\left(R_{>30} \mid \theta_5\right), U\left(\vec{y}_9\right);\ P\left(R_{<30} \mid \theta_5\right), U\left(y_{10}\right) \right\rangle,$$

$$l_{16} = \left\langle P\left(R_{>30} \mid \theta_6\right), U\left(\vec{y}_{11}\right);\ P\left(R_{<30} \mid \theta_6\right), U\left(y_{12}\right) \right\rangle,$$

$$l_{17} = \left\langle P\left(R_{>30} \mid \theta_7\right), U\left(\vec{y}_{13}\right);\ P\left(R_{<30} \mid \theta_7\right), U\left(y_{14}\right) \right\rangle,$$

$$l_{18} = \left\langle P\left(R_{>30} \mid \theta_8\right), U\left(\vec{y}_{15}\right);\ P\left(R_{<30} \mid \theta_8\right), U\left(y_{16}\right) \right\rangle$$

The solution procedure involves firstly the calculation of the expected utility of each simple lottery. Then, these results are combined to calculate the overall expected utility of the FCOL. Following this procedure, (4-5) was derived and implemented in MATLAB.

$$E\left(u \mid L(A_1)\right) = \sum_{i=1,\, j=1}^{i=16,\, j=8} \Big(P\left(\theta_j \mid \vec{x}\right) \times P\left(R_{>30} \mid \theta_j\right) \Big) \times U\left(\vec{y}_i\right) + \ldots \\ \left(P\left(\theta_j \mid \vec{x}\right) \times P\left(R_{<30} \mid \theta_j\right) \right) \times U\left(\vec{y}_{i+1}\right) \tag{4-5}$$

In (4-5): $E\left(u \mid L(A_1)\right)$ is the expected utility of the simple lottery; $P\left(\theta_j \mid \vec{x}\right)$ is the probability of the state, where $j = 1 \ldots 8$ as described by Table 4-10; $P\left(R_{>30} \mid \theta_j\right)$ and $P\left(R_{<30} \mid \theta_j\right)$ were defined previously; and $U\left(\vec{y}_i\right)$ is the multi-attribute utility value of consequence $\vec{y}_i$, where $i = 2j - 1$, also shown in Figure 4-3. The values of the subjective conditional likelihoods used in (4-5) are shown in Table 4-15.

Table 4-15: Subjective Probabilities for $E(u \mid L(A_1))$

STATE OF NATURE θ_j	$P\left(R_{>30} \mid \theta_j\right)$	$P\left(R_{<30} \mid \theta_j\right) = 1 - P\left(R_{>30} \mid \theta_j\right)$
θ_1	0	1
θ_2	1	0
θ_3	0.1	0.9
θ_4	1	0
θ_5	1	0
θ_6	0.5	0.5
θ_7	1	0
θ_8	1	0

4.1.4.2. Lottery Model for A_1, Context 3

Action A_1 when the vessel is at sea (Refer to Table 4-8) was modelled as a simple OL of the form shown in Figure 4-4. In this context, only the state of nature influences the expert's preferences toward the ultimate consequences of the maintenance decision.

$$l\left(A_1\right) = \left\langle \begin{matrix} P\left(\theta_1 \mid \vec{x}\right), U\left(\vec{y}_1\right) \; ; \; P\left(\theta_2 \mid \vec{x}\right), U\left(\vec{y}_2\right) \; ; \; P\left(\theta_3 \mid \vec{x}\right), U\left(\vec{y}_3\right) \; ; \; P\left(\theta_4 \mid x\right), U\left(y_4\right) \; ; \ldots \\ P\left(\theta_5 \mid \vec{x}\right), U\left(\vec{y}_5\right) \; ; \; P\left(\theta_6 \mid \vec{x}\right), U\left(\vec{y}_6\right) \; ; \; P\left(\theta_7 \mid \vec{x}\right), U\left(\vec{y}_7\right) \; ; \; P\left(\theta_8 \mid x\right), U\left(y_8\right) \end{matrix} \right\rangle$$

Figure 4-4: Simple Lottery $l\left(A_1\right)$, Context 3

The expected utility of any simple lottery l_m using can be calculated directly using (4-6) which can be found in references such as (Clemen, 1996). In (4-6): $E\left(u \mid l_m\right)$ is the expected utility of the simple lottery l_m, state of nature probabilities are represented by $P\left(\theta_j \mid \vec{x}\right)$ and the corresponding utilities or MAU values for each consequence are represented by $U\left(\vec{y}_i\right)$. Expected utility solutions of this lottery using (4-6) were determined using MATLAB.

$$E\left(u \mid l_m\right) = \sum_{1}^{j} \left(P\left(\theta_j \mid \vec{x}\right) \times U\left(\vec{y}_i\right) \right) \tag{4-6}$$

4.1.4.3. Lottery models for A_2 to A_5, Contexts 1 - 4

All remaining actions A_2 to A_5 in all contexts (Refer to Table 4-8) were modelled as recursive ICOLs of the general form shown in Figure 4-5. The decision context does not change the structure of the lottery models as A_2 to A_5 are all deferral actions although the subjective probabilities change.

It was assumed that utilities were constant as the time periods modelled by the actions are at maximum 24 hours, and that repetition of the same action follows if the pump does not stop within the timeframe specified by the action.

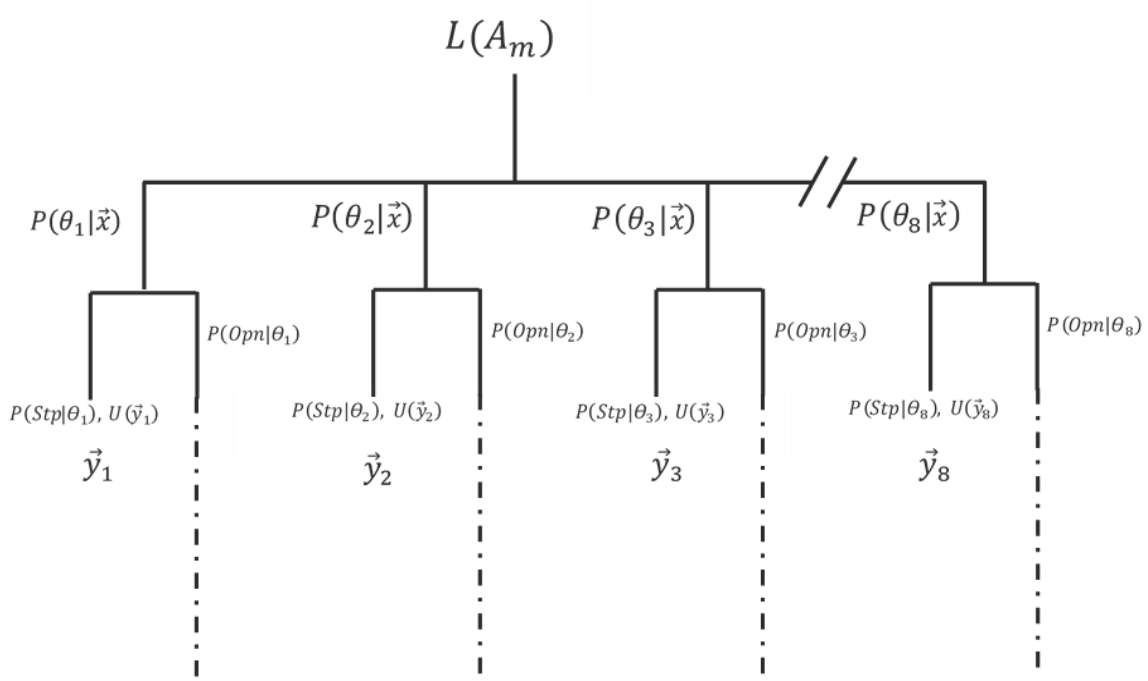


Figure 4-5: Recursive Infinite Compound Lottery $L(A_m)$ where m = 2, 3, 4, 5

Figure 4-5 describes deferral actions which whose outcomes are affected by one of eight states of nature θ_j. Each state has a pair of associated probabilities which will affect whether the pump stops or continues running without obvious faults. The consequences $\vec{y}_i$ are described by these specific probabilities and a multi-attribute value $U(\vec{y}_i)$.

$P(Stp|\theta_j)$ and $P(Opn|\theta_j)$ are required as they influence the expert's maintenance decision-making. These represent the conditional likelihood of the pump stopping and remaining operational throughout the period specified by A_m, given that the state of nature is θ_j. These values were elicited from the expert. $P(Stp|\theta_j)$ is tabulated for all actions in Table 4-16. $P(Opn|\theta_j)$ may be calculated from these values considering the relationship $P(Opn|\theta_j) = 1 - P(Stp|\theta_j)$.

Table 4-16: $P(Stp|\theta_j)$ **used for** $E\left(u|L(A_m)\right)$ **where** $m = 2, 3, 4, 5$

	ICOL			
STATE OF NATURE θ_j	$L(A_2)$	$L(A_3)$	$L(A_4)$	$L(A_5)$
θ_1	0.005	0.005	0.01	0.01
θ_2	0.05	0.1	0.15	0.19
θ_3	0.1	0.15	0.18	0.2
θ_4	0.2	0.25	0.3	0.35
θ_5	0.05	0.1	0.12	0.15
θ_6	0.5	0.5	0.5	0.5
θ_7	0.05	0.08	0.1	0.15
θ_8	0.25	0.3	0.35	0.4

Applying the simplification algorithm in Section 2.2.5.1, one reformulation of the ICOL is the recursive maximum reduced ICOL shown in Figure 4-6.

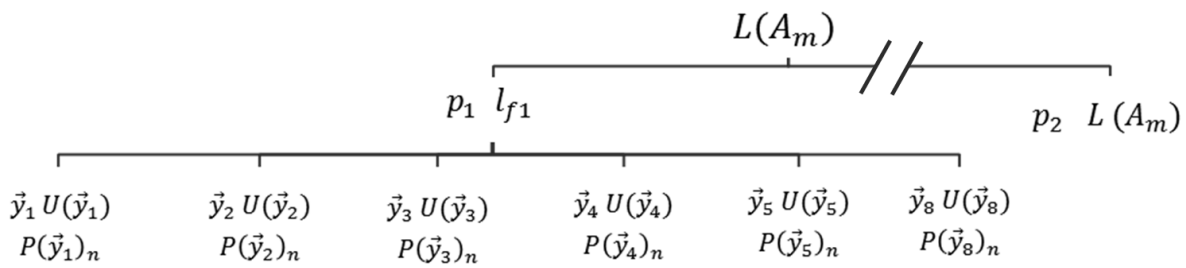


Figure 4-6: Equivalent Representation of Recursive Infinite Compound Lottery $L(A_m)$ **where** $j = 2, 3, 4, 5$

The expected utility of each $L(A_m)$ may be calculated considering this new form. The solution method for recursive maximum reduced ICOLs in Chapter 2 states that only the calculation of $E(u \mid l_{f1})$ is necessary as $E(u \mid l_{f1}) = E(u \mid L(A_m))$. Accordingly, (4-7) and (4-8) were derived to calculate the expected utility of the lottery in Figure 4-6 and implemented in MATLAB.

$$E(u \mid L(A_m)) = E(u \mid l_{f1}) = \sum_{i=1}^{8} \left(P(\vec{y}_i)_n \times U(\vec{y}_i) \right) \tag{4-7}$$

$$P(\vec{y}_i)_n = \frac{P(\theta_j \mid \vec{x}) \times P(Stp \mid \theta_j)}{\sum_{i=1}^{8} \left(P(\theta_j \mid \vec{x}) \times P(Stp \mid \theta_j) \right)} \tag{4-8}$$

The equation (4-8) demonstrates how the probabilities $P(\vec{y}_i)_n$ within the simple lottery l_{f1} are normalised by the sum of the original probabilities of the outcomes $\vec{y}_i$ in Figure 4-5.

4.1.5. CM Data Input

Ongoing use of the maintenance system necessitates a function which either interacts with sensors on the equipment to automatically retrieve and process data or prompts the user to input the processed data. In the present study, the entire CM dataset was analysed at once as all data had already been obtained. The results of this analysis are presented in Chapter 5.

4.2. Stage 4: *Storage and Transmission*

The maintenance scheduling data from the pump application is stored with the present authors as no storage or transmission capabilities exist.

4.3. Performance Measurement: *Availability and Overall Maintenance Cost*

Performance measurement was discussed briefly in Chapter 1. Maintenance strategies such as the present system and periodic PM can be compared using availability and overall maintenance cost. In Chapter 5, the maintenance schedules resulting from the system recommendations and periodic PM may be compared during the pump monitoring period. Availabilities may be calculated using (4-9) and overall maintenance costs calculated as the sum of tools, parts and labour. Alternative metrics such as Mean Time to Repair or Mean Time to Failure (Ebeling, 2005) were not considered due to additional data requirements.

$$Availability = \frac{Uptime}{Uptime + Downtime} \tag{4-9}$$

4.4. Summary

Chapter 4 completed the description of the development and performance measurement of a maintenance scheduling system which began in Chapter 3. The application of a Number 2 General Service Pump aboard a naval tug which was described in the previous Chapter 3, alongside the first two stages of a four-stage system workflow. These were the User Prompt and Data Collection and Processing stages. The present Chapter 4 described the remaining two stages which were Data Analysis and Storage and Transmission.

The User Prompt stage obtains the relevant decision context. Then, in the first part of the Data Analysis stage a relevant linear discriminant classifier trained in labelled pump fault data interprets pump CM data into probabilities of failure. The performance of the set of classifiers was estimated using confusion, certainty and doubt matrices. Confusion and doubt increased while certainty decreased when additional Gaussian noise was added to learning samples to produce test classification vectors. Test vectors were obtained by adding 10%, 50% and 100% Gaussian noise about the class mean to each learning sample in the three different tests. Then, in

the second part of the Data Analysis stage, these probabilities of failure are used within a relevant decision tree containing ICOLs to calculate a maintenance Policy.

The vessel's Chief Engineer is the DM and expert in the present work. Accordingly, he guided the development of the decision trees and lottery models. The experience of this expert was quantified within a MAU function and was employed to estimate values of subjective probability for scenarios which could not be measured.

The completed maintenance system could then be used to analyse the pump CM data and produce a maintenance schedule. Performance measurement of the system against the current periodic PM schedule for the pump shall be conducted using availability and overall maintenance cost; consisting of tools, parts and labour used for a task. These results are presented and discussed in Chapter 5.

CHAPTER 5 - Performance Measurement Results and Discussion

This Chapter primarily presents and discusses the performance measurement of the RBM system developed in the previous Chapters 3 and 4 for the shipboard pump. The expert was asked to recommend a maintenance action if each ‘fault found’ or ‘no fault found’ state was certain to occur, while system was used to compute maintenance actions given the same certain states. These recommendations were compared and discussed. The RBM schedule generated by the analysis of pump CM data is also compared with the current periodic PM schedule using calculations of availability and overall maintenance cost.

In the second part of the Chapter, decision-making using all state probabilities is discussed. This can only be achieved using the RBM system. The role of the system in future predictive maintenance systems is then outlined. Further additional limitations of the pump system are presented and the Chapter concluded with a summary.

5.1. Performance Measurement for Shipboard Pump Application

The Policies calculated using the system can be discussed in two cases. The first case is using probabilities which indicate one state j only is certain to occur, such that $P\left(\theta_j \mid \vec{x}\right)=1$. The second involves probability information describing multiple states simultaneously, such that $P\left(\theta_j \mid \vec{x}\right)<1$ for all j. In Section 4.1.2.2, there were at most five possible actions depending on the decision context. As the decision trees directly linked the decision to an action or deferral, each action corresponds directly to a Policy. For example, the calculation of Policy 1 represents the decision to take Action 1 as it possesses the maximum expected utility of all actions.

Maintenance scheduling according to expert assessment considers only the first case, as a person cannot decide considering multiple states simultaneously without prioritising the highest probability. However, an algorithm such as a decision tree can. The Policies recommended by the expert in the first case and calculated by the system in both cases in are discussed in the following sections.

5.1.1. Maintenance Policies for Certain States

The expert was surveyed to obtain their recommended maintenance Policies in the certain state case and Context 1. Maintenance Policies were also calculated using the RBM system for the same certain state case and Contexts 1-4. Both the expert and the system decided between five possible Policies corresponding to the number of the maintenance actions defined in Chapter 4, Table 4-8 and Table 4-9. The eight states were defined in Chapter 4, Table 4-10 as either a ‘no fault found’ state θ_1 or one of seven ‘fault found’ states, θ_2 … θ_8. Table 4-8, Table 4-9 and Table 4-10 are reproduced below. The four decision contexts were also defined previously in Chapter 4, Section 4.1.2.1. Each described the environment in which the pump operation or maintenance is to be carried out, such as ‘Context 4 - Vessel at Sea, Emergency’. The Policy results are displayed in Table 5-1.

[Reproduced] Table 4-8: Maintenance Actions for Number 2 General Service Pump, Contexts 1-3

NUMBER	ACTION	
	Context 1-2	***Context 3***
A_1	Stop No.2 Pump to check inside	Stop No. 2 Pump to check inside, change to No. 1 Pump immediately
A_2	Inspect pump more than once in 6 hours	Inspect pump more than once in 6 hours
A_3	Inspect pump after 6 hours	Inspect pump after 6 hours
A_4	Inspect pump after 12 hours	Inspect pump after 12 hours
A_5	Inspect pump after 24 hours / Do nothing	Inspect pump after 24 hours / Do nothing

[Reproduced] Table 4-9: Maintenance Actions for Number 2 General Service Pump, Context 4

NUMBER	ACTION
	Context 4
A_2	Inspect pump more than once in 6 hours
A_3	Inspect pump after 6 hours
A_4	Inspect pump after 12 hours
A_5	Inspect pump after 24 hours / Do nothing

[Reproduced] Table 4-10: States of Nature for Number 2 General Service Pump

CLASS NUMBER c_k	**STATE OF NATURE** $P(\theta_j \mid \vec{x})$	**DESCRIPTION**
1	$P(\theta_1 \mid \vec{x})$	No Fault Found
2	$P(\theta_2 \mid \vec{x})$	Worn Impeller
3	$P(\theta_3 \mid \vec{x})$	Loose Packing
4	$P(\theta_4 \mid \vec{x})$	Damaged Pump Drive-End Bearing
5	$P(\theta_5 \mid \vec{x})$	Worn Pump Drive-End Bearing
6	$P(\theta_6 \mid \vec{x})$	Loose Pump Foot
7	$P(\theta_7 \mid \vec{x})$	Static Imbalance in Shaft
8	$P(\theta_8 \mid \vec{x})$	Offset Misalignment in Shaft

Table 5-1: Certain State Policy Number Comparison

	Policy Number				
Certain State j $P(\theta_j \mid \vec{x}) = 1$	***Expert, Context 1***	***System, Context 1***	***System, Context 2***	***System, Context 3***	***System, Context 4***
1	5	5	5	1	5
2	3	5	5	1	5
3	1	1	1	1	5
4	1	1	1	1	5
5	3	1	1	1	5
6	1	5	5	1	5
7	1	1	1	1	5
8	1	1	1	1	5

Table 5-1 shows that the system prefers to abstain from inspections in the case of a worn impeller ($P(\theta_2 \mid \vec{x}) = 1$), perform maintenance for the worn bearing case ($P(\theta_5 \mid \vec{x}) = 1$) and abstain also for the loose foot fault ($P(\theta_6 \mid \vec{x}) = 1$) in comparison to the expert in Context 1.

$P(\theta_2 | \vec{x})$ represents a worn impeller. Replacing the impeller requires that the entire pump is dismantled. Decisions to abstain from inspections seem logical provided the pump is still performing to an adequate standard.

$P(\theta_5 | \vec{x})$ represents a 'worn pump drive-end bearing' fault. The drive-end bearing supports and aligns the pump-motor shaft which is crucial to the operation of the pump. Conducting maintenance also seems logical in this case although the pump must be dismantled to replace the bearing, repair time is approximately 30 minutes and the bearing costs approximately $20 AUD.

$P(\theta_6 | \vec{x})$ represents the 'loose pump foot' fault condition, which could lead to loosening of other feet if not addressed as soon as is practical. Symptoms of this fault are reflected as changes in the vibration of the pump. The system considers that the pump does not require maintenance immediately in this state as it in itself is not a significant issue while the vessel is alongside, provided that the pump continues to perform adequately. However, a loose foot has the potential to cause misalignment of the pump which the system suggests requires immediate maintenance.

Table 5-1 also shows that the context selected reasonably affects how the system determined the Policy. In Context 3, the system recommends maintenance in all cases as it is more vigilant while the vessel is moving and there is the opportunity to stop the pump, even though this does not strictly make sense in the context of the 'no fault' state. In future studies, a longer duration of learning data collection for this class is recommended as a potential remedy. This sensitivity was shown in the 'holdout' tests in Chapter 4, Section 4.1.1.2. C3/C4 has the largest performance reduction when noise in the test data was increased. The expected utility solutions between Policies 1 and 5 had a difference of 0.001003 or approximately 0.01%.

In Context 4, it can be observed that the system recommends running the pump until the next day in all cases as there is no opportunity to stop the pump in an emergency.

5.1.2. Maintenance Policies for Shipboard Pump versus Periodic PM

The value of the RBM system can be demonstrated with a high-level comparison between periodic PM and system recommendations for the Number 2 General Service Pump. This is achieved using calculations of availability and overall maintenance cost over the CM data collection period.

CM data were collected as described in Chapter 3, Section 3.5.2 between 15/03/2018 and 2/08/2018 and processed by the system into Policies. Within this period, the pump was run at $20m^3hr^{-1}$ for approximately 780 hours to obtain a greater number of operational hours and maintenance events.

The calculated Policies, periodic PM and actual maintenance work done during this period are presented in Table 5-2. Two methods of calculation are presented; Policies calculated without apriori probabilities (Policy N/W), and Policies calculated where all apriori probabilities are assumed to be equal (Policy P/W). Estimates of apriori probabilities could not be obtained for the present application due to lack of historical data on the ‘fault found’ conditions used in the present study. The maintenance done on each CM collection date was determined using the pump maintenance record over the CM period which can be found in Appendix A. Calculation methods are described further in Appendix C.

Table 5-2: System Policies, Periodic PM and Maintenance Performed

DATE	POLICY N/W	POLICY P/W	PERIODIC PM	MAINTENANCE PERFORMED
15/03/2018	1,1,1,1,1	1,1,1,1,1	5	
27/03/2018	1,1,1,1,1	1,1,1,1,1	1	Inspected, repair not required
12/04/2018	1,1,1,1,1	1,1,1,1,1	5	
26/04/2018	1,1,1,1,1	1,1,1,1,1	1	Inspected, repair not required
9/05/2018	1,1,1,1,1	1,1,1,1,1	5	
25/05/2018	1,1,1,1,1	1,1,1,1,1	5	
7/06/2018	1,1,1,1,1	1,1,1,1,1	1	Performed repair
21/06/2018	1,5,5,1	5,5,5,5	5	
5/07/2018	1,1	1,1	1	Performed repair
19/07/2018	1,1,1,1,1	1,1,1,1,1	5	
2/08/2018	1,1,1,1,1	1,1,1,1,1	5	

Considering the Policies which did not account for the apriori probabilities (Policy N/W) during the CM period, it can be observed that the system recommends maintenance most of the time, which does not result in a reduction in availability or overall maintenance cost when compared to the periodic PM schedule. Also, fewer 'perform maintenance' results (Policy = 1) were obtained when apriori probabilities were involved in the calculations (Policy P/W). The inclusion of the apriori probabilities improves the Policy selection as a period of good pump condition was observed following actual maintenance on 7/06/2018, whereas the exclusion of the priors meant that this period was not identified in all samples. It is recommended that good estimates of the apriori probabilities are obtained to improve Policy selection. Table 5-2 also shows that only Policies 1 and 5 are selected by the system while 2,3 and 4 are not. Selection of Policies 2,3 and 4 is explored in the following Section 5.2.

It is possible that the pump bearing required more maintenance than was done for optimal performance, that corrosion affected learning and CM data collection, and that the 'worn bearing' learning dataset's quality was inadequate. These factors as well as the following additional calculation data in Table 5-3 are discussed as follows to explain the results in Table 5-2.

Table 5-3: System Posterior Probabilities, Polices, Periodic PM and Maintenance Performed

DATE	$P(\theta_1 \mid \vec{x})$		$P(\theta_5 \mid \vec{x})$		POLICY P/W	PERIODIC PM	MAINTENANCE PERFORMED
	MIN	MAX	MIN	MAX			
15/03/2018	7.66E-141	8.74E-106	1	1	1,1,1,1,1	5	
27/03/2018	3.37E-120	7.2E-102	1	1	1,1,1,1,1	1	Inspected, repair not required
12/04/2018	3.78E-175	1.17E-153	1	1	1,1,1,1,1	5	
26/04/2018	1.55E-191	5.67E-177	1	1	1,1,1,1,1	1	Inspected, repair not required
9/05/2018	8.19E-144	7.01E-83	1	1	1,1,1,1,1	5	
25/05/2018	4.03E-101	4.87E-79	1	1	1,1,1,1,1	5	
7/06/2018	4.16E-15	1.05E-07	1	1	1,1,1,1,1	1	Performed repair
21/06/2018	1	1	1.26E-32	2.04E-13	5,5,5,5	5	
5/07/2018	2.55E-18	2.19E-14	1	1	1,1	1	Performed repair
19/07/2018	1.47E-92	2.2E-73	1	1	1,1,1,1,1	5	
2/08/2018	3.06E-112	1.59E-94	1	1	1,1,1,1,1	5	

The probability data in Table 5-3 clarifies why the system frequently recommended maintenance. The approximate certain state condition is generally evident for $P(\theta_5 \mid \vec{x})$, and for $P(\theta_1 \mid \vec{x})$ on 21/06/2018. The system is consistent and has produced the same recommendations as those shown previously in Table 5-1.

The most important data in Table 5-3 are the probabilities and Policies corresponding to the dates on which maintenance was done during the CM period (Maintenance Performed = Performed repair). The system identified correctly that the pump drive-end bearing maintenance was necessary with high probability on bearing wear, which aligns with the actual maintenance done on 07/06/2018. However, the system could not identify the pump was experiencing an issue on 05/07/2018 as the pump packing had been tightened prior to the CM data collection, and the issue may not have been similar enough to the experimental packing fault or presented itself early enough to be able to be detected on other dates.

The interference which may have been provided by corrosion in both the test pump and the Number 2 General Service Pump should not be ignored. Contrary to what may seem sensible in this application, neither of these pumps are designed for marine applications. Both test and

Number 2 General Service Pump were identical albeit operated differently. Both pumps were often left idle – meaning that the cast iron interior of each pump reacted to the salt and severely corroded. This corrosion may have affected the vibrational characteristics of the pumps and their components, possibly having contributed to the frequency measurement errors of up to 4.5Hz previously shown in Table 3-11.

For the case of an inadequate 'worn bearing' learning dataset, the use of the air conditioning pump in Section 3.6.2 was discussed, suggesting that resource limitations of the study prevented an improved wear testing approach. It was recommended that improved wear testing methods are used in future studies. Given the results in Table 5-3, it is also possible that the air conditioning pump's bearing may not contain a large degree of wear, explaining why it is often identified as the dominant probability in place of the 'no fault found' condition.

The present pump RBM system can perform a comprehensive analysis of pump CM data and determine the best possible maintenance Policy. The present section has discussed how the calculated maintenance Policies are reasonable though these results were affected by several factors.

The theoretical results thus far indicate that the system recommendations do not provide an increase in availability and corresponding reduction in overall maintenance cost versus periodic PM, and some reasons for this discrepancy were identified as primarily data quality issues. It is expected system recommendations shall align more with the maintenance done in future studies which adopt the present methodology and address the issues discussed previously. Future studies would also benefit from testing their application using a true comparison as opposed to the present 'hindsight' approach, which was the best possible comparison given time and resource limitations and the substantial degree of risk involved in conducting the present study. A true comparison would involve maintenance of the Number 2 General Service Pump being performed in line with the system's recommendations. It is strongly recommended that the present system is used in a future work to compare availability and maintenance cost to quantitatively demonstrate

its use in this manner, described as part of the operational trial phase of the implementation plan in Chapter 1.

5.2. Managing Ambiguous States

The system recommendations given certain states were presented and compared with an expert in Section 5.1.1. In the present section, the capability of the system to make decisions incorporating multiple state probabilities simultaneously is demonstrated. State probability combinations which result in each Policy are also presented, as not all Policies were calculated during the analysis of the pump CM data. 100 sets of probabilities $P\left(\theta_j \mid \vec{x}\right)$ for $j = 1...8$ were randomly generated and decision analyses performed considering all Contexts 1-4 to calculate all Policies. The results of these analyses are shown in Table 5-4.

Table 5-4: Random Probability Decision Analysis Tests

Context	$P(\theta_1 \mid \vec{x})$	$P(\theta_2 \mid \vec{x})$	$P(\theta_3 \mid \vec{x})$	$P(\theta_4 \mid \vec{x})$	$P(\theta_5 \mid \vec{x})$	$P(\theta_6 \mid \vec{x})$	$P(\theta_7 \mid \vec{x})$	$P(\theta_8 \mid \vec{x})$	Policy N/W
1	0.1229	0.1921	0.0419	0.0183	0.0616	0.0183	0.1088	0.4362	1
1	0.3499	0.0155	0.1470	0.0637	0.2297	0.0285	0.0035	0.1622	1
1	0.0789	0.0751	0.0335	0.2124	0.1107	0.0280	0.4212	0.0402	4
1	0.0610	0.0661	0.2357	0.0644	0.1377	0.0758	0.2961	0.0631	4
1	0.2404	0.1164	0.1350	0.1244	0.0648	0.1178	0.1852	0.0160	5
1	0.2168	0.1369	0.0772	0.1702	0.2566	0.0731	0.0120	0.0571	5
2	0.0815	0.0657	0.1015	0.0063	0.2021	0.0610	0.2281	0.2538	1
2	0.0951	0.2394	0.1796	0.0474	0.1428	0.0701	0.1036	0.1219	1
2	0.0671	0.1606	0.0902	0.1954	0.0503	0.1462	0.2790	0.0112	4
2	0.0420	0.0318	0.0724	0.0338	0.2844	0.0953	0.3768	0.0634	4
2	0.2451	0.0802	0.1065	0.0979	0.1573	0.0840	0.1633	0.0657	5
2	0.0333	0.0167	0.0656	0.2153	0.2978	0.1650	0.1242	0.0822	5
3	0.2955	0.1079	0.0614	0.1974	0.0402	0.0708	0.0072	0.2196	1
3	0.0241	0.0902	0.1306	0.2300	0.2002	0.2132	0.0297	0.0820	1
4	0.0309	0.0151	0.1213	0.0792	0.0782	0.2951	0.1009	0.2794	2
4	0.1419	0.0005	0.1310	0.1377	0.0360	0.1484	0.0664	0.3381	2
4	0.1976	0.0156	0.0310	0.0364	0.2188	0.1289	0.0070	0.3647	3
4	0.1299	0.0231	0.1346	0.0146	0.0972	0.1645	0.1160	0.3201	3
4	0.1074	0.1135	0.1724	0.0816	0.1497	0.0611	0.2742	0.0402	4
4	0.1670	0.1156	0.0736	0.0086	0.0297	0.2483	0.3000	0.0571	4
4	0.1620	0.1501	0.1935	0.0302	0.0656	0.1349	0.1434	0.1202	5
4	0.0446	0.1326	0.2399	0.1666	0.0271	0.0037	0.1657	0.2198	5

Table 5-4 demonstrates the combinations of probabilities and context required to obtain Policies 2,3 and 4. Policy 3 is rare, though it can be found using an average of 2000 sets of randomly generated probabilities and Context 4. Some notable examples are also highlighted where the dominant probability suggests that the system should choose a deferral Policy based on the certain state case (Policy = 2,3,4 or 5) and does not, and other examples are highlighted were the dominant probability suggests that the system should select doing maintenance based on the certain state case (Policy = 1) and it does not.

Often, more than one state probability is important in decision-making and therefore all probability information must be incorporated. The previous analysis showed that the system can

manage this type of situation to make a decision. This contrasts with the human capacity to make decisions based on only one (usually the most probable) state. Eight states were considered in the present Thesis although the present RBM methodology can be adapted to any natural number.

5.3. Predictive Maintenance

While not included in the present study, iterative use of the completed maintenance system will support the development of a predictive maintenance system for the given application. The present CM data could also be used to begin to develop the predictive system. The potential role of the present system within a predictive maintenance system is detailed below.

As part of MVA as described previously in Chapter 2, Section 2.1.3.2, a discriminant function for each class $g_i(\vec{x})$ is developed and encloses each class. The probabilities of each state are determined following the evaluation of each discriminant function. It is not straightforward to determine the probability values over time as these will change although must continue to sum to one throughout time to be considered probabilities. $g_i(\vec{x})$ is easier to calculate instead as its value is not constrained , and the existing pump software can be modified to produce $g_i(\vec{x})$. Using regression methods, an approximations of each $g_i(\vec{x})$ over time could be developed using existing CM data. Additional CM data should also be gathered to improve the accuracy of the predictive model. Then, $g_i(\vec{x})$ for a point in the future could be determined using the model, and the future probabilities calculated. These could then be used to predict a maintenance Policy.

Points in the future where $g_2(\vec{x}) \ldots g_8(\vec{x}) > g_1(x)$ are of interest as this would provide a warning of a future fault, given that $g_1(\vec{x})$ is the discriminant function for the 'no fault found' class in each classifier.

A future study using this approach could then determine how well the predicted values match the present maintenance done data in Table 5-2, or alternatively conduct their own investigation, monitoring the pump or another piece of equipment.

5.4. Limitations

The limitations of the present application are discussed briefly within the present section, beyond those assumptions and limitations relating to its scope (Refer to Section 1.6). The limitations of the overall methodology cannot yet be established as the present RBM methodology is at the 'proof of concept' stage in its development.

- It was commented previously that pumps are not strictly a worthwhile application for an advanced maintenance system, however they are a fundamental component of any mechanical system and therefore a useful starting point. The methodology is intended for use on equipment which would otherwise require numerous personnel hours spent monitoring the system. In some cases, this may include pumps.
- Large volumes of data were not considered due to the pump application which did not require large volumes of data in its development. Similarly, means of secure data storage and management were not considered. These requirements should be addressed in line with the development of future applications.
- Data for use with a supervised classification algorithm to achieve the best possible performance is generally difficult to obtain due to its cost and time requirements. Chapter 2 highlighted that the selection of a suitable algorithm contributes toward cost management, although resource limitations imposed on the present study meant that only ten 'no fault found' or 'fault found' datasets could be obtained. Future pump applications would benefit from an improved 'worn bearing' learning dataset. It is expected that the present approach to obtain this dataset using the air conditioning pump has affected the system's performance.
- Expert guidance beyond the elicitation of utilities and subjective probabilities was required to develop accurate lottery models, and this consultation is a time-consuming and iterative process. Software developed within the present Thesis shall guide the development of future systems, although expert knowledge and guidance shall remain critical to an application's success and an important part of its development.

- Due to time limitations and the inherent risk involved, an operational trial could not be conducted. The ideal case would have been to perform maintenance in line with the system recommendations, allowing for a direct availability and overall maintenance cost comparison. Further work is necessary beyond the scope of the present Thesis, and the operational trial step is included in the equipment-level implementation plan previously in Chapter 1.
- The specific system developed within the present work is limited to the analysis of the Number 2 General Service Pump, and likely pumps in similar working conditions like the Number 1 General Service Pump. Even then, it is necessary to re-assess the system parameters and possibly modify the system software. It is not reasonable to assume that all pumps are created equal as there are a large variety of pump applications aboard a vessel, which each require a different configurations. Variations would be reflected in the appropriate experimental and CM data, and possibly how the pump is maintained. Developing RBM systems for different pump configurations is suggested as a future work.
- The present RBM methodology does not consider sub-systems, though future applications to sub-systems are described in Chapter 1 and are intended as a future work.
- The present system does not predict trends in equipment condition, although repetitive use of the system may be used to produce this information over time as discussed previously. However, a predictive system will require additional CM data as it will guarantee improved system accuracy.

5.5. Summary

Chapter 5 presented and discussed results generated by the RBM system for the Number 2 General Service Pump and discussed the additional capabilities and possibilities of the system. Initially, single states were considered and the systems' recommendations did not strictly align with the expert DM. The RBM schedule produced using CM data was then compared to the pump periodic PM schedule. The comparison indicated that the system did not provide an availability

increase or a corresponding decrease in overall maintenance cost, though it made recommendations which were consistent with the single state case. The maintenance requirements of the pump and the wear classification learning dataset were then discussed as some of the inhibiting factors, as well as the fact that the CM data were not gathered under true comparison conditions of maintenance performed according to system recommendations. This was due to the degree of risk involved. Additional Policy results calculated using randomly generated probabilities were then presented. These results highlighted cases in which all Policies were calculated, and cases when the dominant probability did not result in the same Policy as the single state case. These additional results emphasised the capability of the system to go beyond human capability to make decisions which involve all relevant states simultaneously without loss of information. The number of states itself is irrelevant as the approach can be developed for any natural number of states. It was then outlined how the present approach may be modified for future use as part of a predictive maintenance system where regression models are created for $g_i(\vec{x})$. Lastly, additional limitations beyond those discussed previously in Section 1.6 were discussed. Most of these can be removed with the contribution of additional knowledge provided by future work which is discussed in the following Chapter.

CHAPTER 6 - Conclusions

The focus of this Thesis was the application of RBM to marine vessel maintenance, demonstrated using a shipboard pump case study. Using, CM, ML and Decision Theory, RBM was expected to determine the optimal maintenance time and best possible Policy to reduce total maintenance cost. The RBM system was developed for a shipboard pump in Chapters 3 and 4. It was applied to determine the optimal maintenance time and the best Policies for the pump in Chapter 5. Additional capabilities of the system were also discussed, alongside its role in future predictive maintenance systems.

The current Chapter presents the conclusions of the present Thesis, followed by recommendations for future projects which further develop the present system and RBM concept.

6.1. Conclusions

The present Thesis has developed RBM methodology and provided an example of how RBM may be applied to determine optimal maintenance timing and the best possible maintenance tasks considering a shipboard pump. In conducting the research for this Thesis; literature was evaluated in the fields of marine vessel maintenance, CM, ML and Decision Theory. Accordingly, the RBM framework and novel approaches in ML and Decision Theory were developed, and an expert was consulted throughout the process. The resulting Thesis contains material which addresses the Research Objectives outlined in Section 1.5. **Research Objective 1** was achieved in Section 1.2.4, **Research Objectives 2, 3** and **4** were achieved in Chapters 3 and 4, and lastly **Research Objective 5** was achieved in Chapter 5.

Considering the outcomes of the research, system development and system performance measurement conducted within the present Thesis, the following has been concluded:

Chapter 1 highlighted that innovation has not occurred in relation to the maintenance of marine vessels as the risk of catastrophic failure is minimal. Therefore, historical practices including periodic PM and RCM were retained. These approaches are not ideal as they cannot determine the optimal time to perform maintenance and disregard that maintenance requirements and schedules should change as equipment ages.

It was necessary to modify the existing RBM framework to include performance measurements, so that the results of maintenance studies using this framework could be easily compared with existing practises. As the marine industry is familiar with comparisons of overall maintenance cost and availability, these should be used to measure the performance of RBM systems developed for marine vessels.

Further discussion in Chapter 1 conclude that the integration and ongoing use of an RBM system requires specialist expertise to ensure consistent development, testing and data management. Software development for each application is as unique as the application, and this unique

process was demonstrated for a shipboard pump in the present Thesis. It would be rare for the software to be re-used, although re-use would reduce the development time for future RBM systems. The considerations in Chapter 1 culminated in an organizational implementation plan for equipment-level RBM applications.

Chapter 2 highlighted that data-driven maintenance has been applied to modernise the maintenance of marine vessels. Discussion of existing studies indicated suitable ML algorithms and outlined the benefits associated with applying Decision Theory to calculate maintenance Policies. Application of these techniques would result in a novel RBM system, as the system would have the capability of calculating the risks of all probable faults and Policies considering all risks.

Chapters 3 and 4 demonstrated how it is possible to build a system with this capability. A supervised Bayesian ML algorithm, Decision Theory and Utility Theory were combined to develop an RBM system for the shipboard pump case study.

Methods and equipment used for data collection and processing were described in Chapter 3. Experimental learning data were required to develop the supervised Bayesian classifiers and CM data were required to calculate RBM Policies. Learning data for the development of the classifiers described seven possible ‘fault found’ and three possible ‘no fault found’ scenarios. These affect the operation and maintenance of the Number 2 General Service Pump. Learning data were obtained by conducting ten tests using an alternate pump which was configured to simulate each scenario.

CM data were obtained from the Number 2 General Service pump while it was running in an ambiguous state which contained all ‘fault found’ and ‘no fault found’ states simultaneously. CM data were collected fortnightly over approximately six months.

Learning and CM data were developed from vibration, temperature, pressure, motor current, tachometer and packing drip rate measurements.

Data from the ten scenarios were used to develop four supervised Bayesian classifiers in Chapter 4. These four classifiers each represented an operational context. Each context indicated the status of the vessel and other equipment while the Number 2 General Service Pump was required to operate. The four contexts included: 'Vessel alongside, no engines running', 'Vessel alongside, engines running', 'Vessel at Sea, no emergency' and 'Vessel at Sea, emergency'.

It was necessary to use the supervised Bayesian classifiers to produce probabilities as the probabilities could be used to fully quantify risk. No tools have existed prior to the present Thesis which could be used to interpret probabilities and estimate the performance of the classifiers. To address this issue, a new approach was presented in Chapter 2. This approach consisted of applying the confusion, certainty and novel doubt matrices which each computed different properties of a supervised Bayesian classification algorithm. This approach was later applied to estimate the performance of the supervised Bayesian classifiers developed for the shipboard pump. A similar approach was also presented to adapt the certainty and doubt matrices for non-Bayesian algorithms as non-Bayesian algorithms cannot produce probabilities.

The 'hold-out' performance of the classifiers was estimated using noisy data based on the learning samples in Chapter 4. Probabilities were interpreted using confusion, certainty and doubt matrices which demonstrated that the classifiers were ~100% accurate with ~100% certainty and ~0% doubt without noise. Accuracy and certainty declined as the noise in the input data increased between levels of 10%, 50% and 100%.

Decision Theory and Utility Theory are useful tools in the context of maintenance decisions and can be applied to calculate a viable Policy considering the probabilities of all faults relevant to a piece of equipment. Probability information could be obtained using supervised Bayesian classification algorithms, which was the approach used in the present Thesis. The theories outlined methods to measure internal qualities such as preference and subjective probability estimates. The Policies calculated aligned with a rational DM's preference despite the fuzzy-rational behaviour of an expert DM whose data have been used within the decision analysis.

The parameters of the pump decision analysis were defined for the RBM system in Chapter 4. One decision tree was created for each of the four pump classifiers previously described and most of the maintenance actions within these trees were maintenance deferral actions, which could be modelled using ICOLs.

While the current theories provided ways to model uncertain outcomes as lotteries, they did not provide solution methods for ICOLs. A novel simplification algorithm and solution method for the special recursive case was developed in Chapter 2 and both were applied to determine the expected utility solutions of the ICOLs in Chapter 4. Thus, a maintenance Policy could be calculated from the trees.

The theoretical system recommendations discussed in Chapter 5 resulted in lower availability and higher maintenance costs than the prescribed PM schedule over the CM period. Reasons for this discrepancy between periodic PM and the system recommendations were identified as one or a combination of: the pump requiring more maintenance than was done for optimal performance, or an inadequate learning dataset.

The system should be used in future operational trials so that the recommendations of the system can be followed to obtain true availability and overall maintenance cost comparisons. The work conducted in the present Thesis demonstrated the development of a maintenance system and monitoring of the pump as opposed to conducting maintenance according to the system recommendations.

Chapter 5 discussed the unique ability of the system to make decisions in ambiguous situations beyond human capacity, and how all probabilities should be used to make decisions as opposed to the assumption that the most probable class is certain to occur. This was demonstrated concisely using test Policies calculated from randomly generated probabilities. Some of these test Policies contradicted the action suggested by the expert although the dominant probability was the same as the probability considered by the expert whom only considers one risk.

Chapter 5 also discussed how the present system can be modified and used as part of a future predictive maintenance system. A predictive regression model can be derived from the existing system and would provide more accurate predictions with additional CM data.

The present Thesis has identified and addressed some of the issues surrounding the maintenance of marine vessels and developed a RBM methodology which is also applicable in other industries desiring to adopt a more efficient, data-driven maintenance approach. The intricacies of the RBM system developed were discussed in detail. The present Thesis provides a strong foundation for future work toward the development of extraordinary maintenance.

6.2. Future Work

Numerous avenues for future work beyond the scope of the present Thesis have been identified and are described below. Small-scale projects describe improvements to the present work, while medium and large-scale projects are novel in some cases though all are extensions to the present RBM methodology. Lastly, the completion of all small-scale projects is the foundation for successful medium-scale projects and the medium-scale projects are the foundations of the large-scale projects.

Small-scale

- It has been outlined how the present system could be used in predictive maintenance for the present shipboard pump. In a future study, the present system should be modified to obtain data necessary to develop a predictive maintenance model over a period of at least six months as this corresponds to a minimum of two occurrences of maintenance work. A key outcome of the predictive maintenance investigation would be to determine the minimal number of CM measurements required to develop a regression model which predicts maintenance within the CM time with reasonable accuracy, high certainty and low doubt.
- It is worthwhile to develop a method to manage data collection and storage to ensure quality CM datasets, which can be used with the present pump system. The present application

involved manual data collection for some measurements and this is not practical for long-term applications greater than six months or higher volume data collection regimes.

- While the system performance was logical, each learning dataset was limited to a single and potentially extreme version of a fault present within the pump. Obtaining additional experimental datasets is worthwhile to describe the progression of all faults for the present pump, combined with modification of the present system to utilize them in classification and decision-making. Further, the system will benefit from the development of an improved wear testing approach to obtain wear data from shipboard components such as the present pump as well as bearings and shafts.
- The apriori probabilities in the present application were assumed to be equal as no better estimates were available. The results indicated that this assumption improved the recommendations of the system regarding the identification of maintenance needs shown by the actual maintenance done. It is recommended that improved estimates of these apriori probabilities are obtained for all future applications, and further that these estimates are required prior to any application proceeding to operational trials and permanent integration. Possible means to obtain these estimates include a series of run-to-failure experiments for the specific application or a similar piece of equipment or provision of the data from the OEM.
- MVA is not the only classification approach which can be used to create an RBM system. It is worthwhile to investigate other ML algorithms or alternative probability estimation methods, as this defines the data collection regime and application cost as discussed in Chapter 2.

Medium-scale

- One possible extension of the present application is an investigation into the applicability of the present system without modifications to a variety of pump configurations for both diagnosis and predictive maintenance. Some measure of similarity to the original pump configuration could then be developed to describe the other configurations. This investigation

has value as often only one piece and type of equipment shall be made available for experimental work and testing.

- Another possible extension of the present system is an investigation of the failure behaviour of similar pumps or alternatively other equipment in series and in parallel combinations to create a sub-system application for both diagnosis and predictive maintenance. This is the next logical step in extending the system toward full-vessel implementation where its full potential can be realised. In this investigation, it will be necessary to determine how the failure or progression toward failure of the first component affects the second, and to determine this relationship for all possible failure modes beyond the concept of reliability. However, reliability may prove a useful foundation. This shall be an experimental investigation as no single piece of equipment can be fully described by analytical failure mode equations.
- A third possible extension of the present system is the creation of a maintenance system for other pieces of equipment such as generators, gas turbines and propeller shafts according to the methodology proposed in the present Thesis and for predictive maintenance. This investigation can be conducted following the process outlined in the present Thesis where failure modes are identified, experiments conducted, and software developed using linear discriminant classifiers and decision trees. The resulting software can then be compared to any existing application in terms of development time, cost and complexity.

Large-scale

- Once the knowledge of how component failure relationships in series and parallel exists in tandem with a fully developed and tested component maintenance system, these can be integrated into a sub-system application. The methodology itself shall then require further development to guide the development and integration of sub-system applications. Again, a foundation for this may be based in reliability concepts or other concepts depending on how a

component’s failure or progression toward failure affects components connected in series or parallel.

- The final stage of RBM methodology development concerns its use for a complete vessel, demonstrated with an application and operational trial. The methodology itself will not necessarily require an overhaul but must be adapted to consider the huge datasets involved in CM data collection, processing, management and storage. Emphasis should be placed on developing processes and software which are reliable and secure for marine applications at this stage.
- The successful vessel application and integration shall culminate in a significant contribution to the development of an end-user software package for use with other vessels. All software packages should be compatible with suitable monitoring hardware and data storage facilities. Ideally, software packages would include: a data visualisation module; data and supervised classification algorithms for use with a wide variety of equipment; an algorithm performance estimation module; utility and subjective probability survey modules; decision tree design and evaluation modules; reporting modules, and performance testing modules for hardware and software. The vessel-level package should be user-friendly, informative and tailored to application requirements.
- The commercialisation of the methodology is complete when overarching software and hardware are developed, tested and integrated to provide enterprise-level asset management capability which includes RBM and predictive maintenance for a fleet of vessels. Due to the methodology presented in this Thesis, it shall be possible to observe the progression of individual failure modes within each piece of equipment, to forecast individual part replacements and maintenance skill requirements which shall improve fleet readiness.

References

Ahmadi, A., Gupta, S., Karim, R., & Kumar, U. (2010). Selection of Maintenance Strategy for Aircraft Systems using Multi-Criteria Decision Making Methodologies. *International Journal of Reliability, Quality and Safety Engineering, 17*(03), 223-243.

Ariely, D. (2008). *Predictably irrational*. New York, USA: HarperCollins.

Arunraj, N., & Maiti, J. (2007). Risk-based maintenance—Techniques and applications. *Journal of Hazardous Materials, 142*(3), 653-661.

Baliwangi, L., Ishida, K., Arima, H., & Artana, K. B. (2006). *Optimizing Ship Machinery Maintenance Scheduling Through Risk Analysis and Life Cycle Cost Analysis.* Paper presented at the 25th International Conference on Offshore Mechanics and Arctic Engineering, Vancouver, Canada.

Ballabio, D., Grisoni, F., & Todeschini, R. (2018). Multivariate comparison of classification performance measures. *Chemometrics and Intelligent Laboratory Systems, 174*, 33-44.

Basurko, O. C., & Uriondo, Z. (2015). Condition-Based Maintenance for medium speed diesel engines used in vessels in operation. *Applied Thermal Engineering, 80*, 404-412.

Bayoumi, A., Goodman, N., Shah, R., Eisner, L., Grant, L., Keller, J., & Center, C.-B. M. (2008). *Conditioned-Based Maintenance at USC-Part IV: Examination and Cost-Benefit Analysis of the CBM Process.* Paper presented at the The American Helicopter Society Specialists' Meeting on Condition-based Maintenance, Hunstville, Alabama.

Beebe, R. S. (2004). *Predictive Maintenance of Pumps Using Condition Monitoring* (pp. 181).

Brenner, W., & Herrmann, A. (2018). An Overview of Technology, Benefits and Impact of Automated and Autonomous Driving on the Automotive Industry. In C. Linnhoff-Popien, R. Schneider & M. Zaddach (Eds.), *Digital Marketplaces Unleashed* (pp. 427-442). Berlin, Heidelberg: Springer Berlin Heidelberg.

Burmester, G., Ma, H., Steinmetz, D., & Hartmannn, S. (2018). Big Data and Data Analytics in Aviation. In U. Durak, J. Becker, S. Hartmann & N. S. Voros (Eds.), *Advances in Aeronautical Informatics: Technologies Towards Flight 4.0* (pp. 55-65). Cham, Switzerland: Springer International Publishing.

Carbonell, J. G., Michalski, R. S., & Mitchell, T. M. (1983). An Overview of Machine Learning. In R. S. Michalski, J. Carbonell & T. Mitchell (Eds.), *Machine Learning, Volume I* (pp. 3-23). Palo Alto, California: TIOGA Publishing Company.

Cardona-Morales, O., Avendano, L., & Castellanos-Domínguez, G. (2014). Nonlinear model for condition monitoring of non-stationary vibration signals in ship driveline application. *Mechanical Systems and Signal Processing, 44*(1), 134-148.

Chen, L. N., & Toyoda, J. (1989). *Maintenance scheduling based on two level hierarchical structure to equalize incremental risk.* Paper presented at the Power Industry Computer Application Conference, Seattle, USA.

Cicek, K., & Celik, M. (2013). Application of failure modes and effects analysis to main engine crankcase explosion failure on-board ship. *Safety science, 51*(1), 6-10.

Cipollini, F., Oneto, L., Coraddu, A., Murphy, A. J., & Anguita, D. (2018). Condition-Based Maintenance of Naval Propulsion Systems with Supervised Data Analysis. *Ocean Engineering, 149*, 268-278.

Clemen, R. T. (1996). *Making Hard Decisions* (2, Revised ed. Vol. 0). Pacific Grove, California: Duxbury Press.

Cohen, J. (1960). A coefficient of agreement for nominal scales. *Educational and psychological measurement, 20*(1), 37-46.

Comfort, S., Perera, S., Hudson, Z., Dorrell, D., Meireis, S., Nagarajan, M., . . . Fine, J. (2018). Sorting Through the Safety Data Haystack: Using Machine Learning to Identify Individual Case Safety Reports in Social-Digital Media. *Drug Safety, 41*(6), 579-590.

Coraddu, A., Oneto, L., Ghio, A., Savio, S., Anguita, D., & Figari, M. (2016). Machine learning approaches for improving condition-based maintenance of naval propulsion plants. *Proceedings of the Institution of Mechanical Engineers, Part M: Journal of Engineering for the Maritime Environment, 230*(1), 136-153.

Cordle, J. P. (2017). Revitalizing Shipboard PMS: Out with the MIP, In with the App. *Naval engineers journal, 129*(1), 111-116.

Cullum, J., Nikolova, N., & Tenekedjiev, K. (2019). Expected utility analysis of infinite compound lotteries. *International journal of general systems, 48*(2), 112-138.

de Almeida, A. T. (2007). Multicriteria decision model for outsourcing contracts selection based on utility function and ELECTRE method. *Computers & Operations Research, 34*(12), 3569-3574.

de Almeida, A. T., Ferreira, R. J. P., & Cavalcante, C. A. V. (2015). A review of the use of multicriteria and multi-objective models in maintenance and reliability. *IMA Journal of Management Mathematics, 26*(3), 249-271.

de Melo Brito, A. J., de Almeida Filho, A. T., & de Almeida, A. T. (2010). Multi-criteria decision model for selecting repair contracts by applying utility theory and variable interdependent parameters. *IMA Journal of Management Mathematics, 21*(4), 349-361.

Deputy Assistant Secretary of the Navy. (2019). Highlights of the Department of the Navy FY 2020 Budget. In O. o. Budget (Ed.).

Diamantoulaki, I., & Angelides, D. C. (2013). Risk-based maintenance scheduling using monitoring data for moored floating breakwaters. *Structural Safety, 41*, 107-118.

Dinovitzer, A., Basu, R., Holt, K., Bourne, J., Pegg, N., Hansen, K., & Benckhuysen, J. (1997). A hybrid approach to warship structural maintenance management. Discussion. Authors' closure. *Transactions-Society of Naval Architects and Marine Engineers, 105*, 155-170.

Dong, Y., & Frangopol, D. M. (2015). Risk-informed life-cycle optimum inspection and maintenance of ship structures considering corrosion and fatigue. *Ocean Engineering, 101*, 161-171.

Duda, R. O., Hart, P. E., & Stork, D. G. (2012). *Pattern Classification* (2 ed.). Canada: John Wiley & Sons.

Ebeling, C. E. (2005). *An Introduction to Reliability and Maintainability Engineering*. Illinois, USA: Waveland Press.

Emovon, I. (2016). Ship System Maintenance Strategy Selection Based on DELPHI-AHP-TOPSIS Methodology. *World Journal of Engineering and Technology, 4*(02), 252.

Emovon, I., Norman, R. A., & Murphy, A. J. (2016). An integration of multi-criteria decision making techniques with a delay time model for determination of inspection intervals for marine machinery systems. *Applied Ocean Research, 59*, 65-82.

Eruguz, A. S., Tan, T., & van Houtum, G.-J. (2017). A survey of maintenance and service logistics management: Classification and research agenda from a maritime sector perspective. *Computers & Operations Research, 85*, 184-205.

Farag, Y. B. A. (2017). *A decision support system for ship's energy efficient operation: based on artificial neural network method.* (Master of Science in Maritime Affairs Master), World Maritime University, World Maritime University Dissertations. Retrieved from http://commons.wmu.se/all_dissertations/549

Feili, H. R., Akar, N., Lotfizadeh, H., Bairampour, M., & Nasiri, S. (2013). Risk analysis of geothermal power plants using Failure Modes and Effects Analysis (FMEA) technique. *Energy Conversion and Management, 72*, 69-76.

Ferreira, R. J., de Almeida, A. T., & Cavalcante, C. A. (2009). A multi-criteria decision model to determine inspection intervals of condition monitoring based on delay time analysis. *Reliability Engineering & System Safety, 94*(5), 905-912.

French, S. (1986). *Decision Theory: an introduction to the mathematics of rationality*. Chichester, England: Ellis Horwood Limited, Halsted Press.

French, S., & Insua, D. R. (2000). *Statistical Decision Theory: Kendall's Library of Statistics 9*. Chichester, England: John Wiley & Sons Limited.

Fukunaga, K. (1990). *Introduction to Statistical Pattern Recognition* (W. Rheinboldt Ed. 2nd ed.). San Diego, California, United States of America: Academic Press Inc.

Gao, Z., Cecati, C., & Ding, S. X. (2015). A Survey of fault diagnosis and fault-tolerant techniques—Part II: Fault diagnosis with knowledge-based and hybrid/active approaches. *IEEE Transactions on Industrial Electronics, 62*(6), 3768-3774.

Gholami, B., Phan, T. S., Haddad, W. M., Cason, A., Mullis, J., Price, L., & Bailey, J. M. (2018). Replicating human expertise of mechanical ventilation waveform analysis in detecting patient-ventilator cycling asynchrony using machine learning. *Computers in Biology and Medicine, 97*, 137-144.

Ghosh, D., & Roy, S. (2010). A decision-making framework for process plant maintenance. *European Journal of Industrial Engineering, 4*(1), 78-98.

Giorgio, M., Guida, M., & Pulcini, G. (2015). A condition-based maintenance policy for deteriorating units. An application to the cylinder liners of marine engine. *Applied Stochastic Models in Business and Industry, 31*(3), 339-348.

Gkerekos, C., Lazakis, I., & Theotokatos, G. (2016). *Ship machinery condition monitoring using vibration data through supervised learning.* Paper presented at the International Conference on Maritime Safety and Operations, Glasgow, Scotland.

Goossens, A. J. M., & Basten, R. J. I. (2015). Exploring maintenance policy selection using the Analytic Hierarchy Process; An application for naval ships. *Reliability Engineering & System Safety, 142*, 31-41.

Govindan, K., & Jepsen, M. B. (2016). ELECTRE: A comprehensive literature review on methodologies and applications. *European Journal of Operational Research, 250*(1), 1-29.

Gupta, O., Das, A. J., Hellerstein, J., & Raskar, R. (2018). Machine learning approaches for large scale classification of produce. *Scientific Reports, 8*(1), 5226.

Haase, N., Renkewitz, F., & Betsch, C. (2013). The measurement of subjective probability: Evaluating the sensitivity and accuracy of various scales. *Risk Analysis, 33*(10), 1812-1828.

Handani, D. W., Ishida, K., Nishimura, S., & Hariyanto, S. (2011). *System dynamics simulation for constructing maintenance management of ship machinery.* Paper presented at the IEEE International Conference on Industrial Engineering and Engineering Management (IEEM), Singapore.

Heins, W., & Röling, M. (1995). Application of multiattribute theory in a safety monitor for the planning of maintenance jobs. *European Journal of Operational Research, 86*(2), 270-280.

Huval, B., Wang, T., Tandon, S., Kiske, J., Song, W., Pazhayampallil, J., . . . Cheng-Yue, R. (2015). An empirical evaluation of deep learning on highway driving. *arXiv preprint arXiv:1504.01716*, 1-7.

Islam, R., Abbassi, R., Garaniya, V., & Khan, F. I. (2016). Determination of human error probabilities for the maintenance operations of marine engines. *Journal of Ship Production and Design, 32*(4), 226-234.

Jansen, S. J. (2011). The Multi-Attribute Utility Method. In S. J. T. Jansen, H. C. C. H. Coolen & R. W. Goetgeluk (Eds.), *The Measurement and Analysis of Housing Preference and Choice* (pp. 101-126). Heidelberg, Germany: Springer Netherlands.

Jaspersen, J. G., & Montibeller, G. (2015). Probability Elicitation Under Severe Time Pressure: A Rank ‐ Based Method. *Risk Analysis, 35*(7), 1317-1335.

Kahrobaee, S., & Asgarpoor, S. (2011). *Risk-based Failure Mode and Effect Analysis for wind turbines (RB-FMEA).* Paper presented at the 2011 North American Power Symposium, Boston, Massachusetts, USA.

Kamiel, B. P. (2015). *Vibration-based multi-fault diagnosis for centrifugal pumps*. (Doctor of Philosophy, Engineering), Curtin University, espace - Curtin Institutional Repository. Retrieved from http://hdl.handle.net/20.500.11937/1532

Karydas, D., & Gifun, J. (2006). A method for the efficient prioritization of infrastructure renewal projects. *Reliability Engineering & System Safety, 91*(1), 84-99.

Keeney, R. L., & Raiffa, H. (1993). *Decisions with Multiple Objectives: Preferences and Value Trade-Offs*. Cambridge, England: Cambridge University Press.

Khan, F. I., & Haddara, M. R. (2004). Risk-based maintenance of ethylene oxide production facilities. *Journal of Hazardous Materials, 108*(3), 147-159.

Kingsford, C., & Salzberg, S. L. (2008). What are decision trees? *Nature biotechnology, 26*(9), 1011-1013.

Klein Woud, J., Smit, K., & Vucinic, B. (1997). Maintenance programme design for minimal life cycle costs and acceptable safety risks. *International shipbuilding progress, 44*(437), 77-100.

Klir, G., & Yuan, B. (1995). *Fuzzy Sets and Fuzzy Logic: Theory and Applications* (Vol. 4). New Jersey, USA: Prentice Hall PTR.

Klutke, G.-A., Kiessler, P. C., & Wortman, M. A. (2003). A critical look at the bathtub curve. *IEEE Transactions on Reliability, 52*(1), 125-129.

Kotsiantis, S. B. (2013). Decision trees: a recent overview. *Artificial Intelligence Review, 39*(4), 261-283.

Krishnasamy, L., Khan, F., & Haddara, M. (2005). Development of a risk-based maintenance (RBM) strategy for a power-generating plant. *Journal of Loss Prevention in the Process Industries, 18*(2), 69-81.

Kumar, U. D., Crocker, J., & Knezevic, J. (1999). *Evolutionary maintenance for aircraft engines.* Paper presented at the Reliability and Maintainability Symposium, 1999. Proceedings. Annual., Washington, DC, United States of America.

Lagoudis, I. N., Lalwani, C. S., & Naim, M. M. (2006). Ranking of factors contributing to higher performance in the ocean transportation industry: a multi-attribute utility theory approach. *Maritime Policy & Management, 33*(4), 345-369.

Lazakis, I., Gkerekos, C., & Theotokatos, G. (2018a). Investigating an SVM-driven, one-class approach to estimating ship systems condition. *Ships and Offshore Structures*, 1-10.

Lazakis, I., Raptodimos, Y., & Varelas, T. (2018b). Predicting ship machinery system condition through analytical reliability tools and artificial neural networks. *Ocean Engineering, 152*, 404-415.

Lee, J., Wu, F., Zhao, W., Ghaffari, M., Liao, L., & Siegel, D. (2014). Prognostics and health management design for rotary machinery systems—Reviews, methodology and applications. *Mechanical Systems and Signal Processing, 42*(1), 314-334.

Li, X., Deng, S., Wang, S., Lv, Z., & Wu, L. (2018). *Review of Small Data Learning Methods.* Paper presented at the 2018 IEEE 42nd Annual Computer Software and Applications Conference (COMPSAC), Tokyo, Japan.

Li, Y. H., & Jain, A. K. (1998). Classification of text documents. *The Computer Journal, 41*(8), 537-546.

Ling, H., & Jacobs, D. W. (2007). Shape classification using the inner-distance. *IEEE transactions on pattern analysis and machine intelligence, 29*(2), 286-299.

Lloyd's Register. (2017). Guidance notes for the risk based inspection of hull structures (pp. 38).

Martin, B., McMahon, M. E., Riposo, J., Kallimani, J. G., Bohman, A., Ramos, A., & Schendt, A. (2017). A Strategic Assessment of the Future of U.S. Navy Ship Maintenance: Challenges and Opportunities. Santa Monica, CA, USA: RAND Corporation.

Matthews, B. W. (1975). Comparison of the predicted and observed secondary structure of T4 phage lysozyme. *Biochimica et Biophysica Acta (BBA)-Protein Structure, 405*(2), 442-451.

Merigo, J. M., & Casanovas, M. (2011). A new Minkowski distance based on induced aggregation operators. *International Journal of Computational Intelligence Systems, 4*(2), 123-133.

Mobius Institute. (2015). *Vibration Training Course Book - Category II* (3.1 ed.). Melbourne, Victoria, Australia: Mobius Institute.

Moore Stephens LLP. (2017). Future operating costs report, October 2017.

Moubray, J. (1997). *Reliability-centered Maintenance* (2nd ed.). New York, USA: Industrial Press Inc.

Nedev, A., & Tenekedjiev, K. (1994). Research on the Algorithms for Learning. In A. Nedev & K. Tenekedjiev (Eds.), *Technical Prognostics and Pattern Recognition (In Bulgarian)* (pp. 346 - 358). Varna, Bulgaria: Technical University - Varna.

Nguyen, L. B. (2008). *A multi-attribute decision process for structural material assessment and selection of light-weight, high-performance naval ships.* (Doctor of Science), The George Washington University, The George Washington Univeristy.

Nguyen, V. H. (2017). *Optimal Ship Maintenance Scheduling Under Restricted Conditions and Constrained Resources.* (Doctor of Philosophy, Engineering), Old Dominion Univeristy, ODU Digital Commons.

Nikolova, N., Ahmed, S., & Tenekedjiev, K. (2006). Generic Procedure for Construction of a Multi-Dimensional Utility Function Under Fuzzy Rationality. *International Journal of Computers, Communications & Control, 3*, 422-426.

Nikolova, N., Ahmed, S., & Tenekedjiev, K. (2008a). Generic Procedure for Construction of a Multi-Dimensional Utility Function Under Fuzzy Rationality. *International Journal of Computers, Communications & Control, 3 Supplement S.*, 422 - 426.

Nikolova, N., Mednikarov, B., & Tenekedjiev, K. (2018). Local Risk Proneness in Analytically Approximated Utility Functions Under Monotonically Decreasing Preferences. *Comptes Rendus des Academie Bulgare Des Sciences, 78*(11), 1534-1543.

Nikolova, N., Tenekedjiev, K., Dong, F., & Hirota, K. (2009). Comparison of linear interpolation and arctan approximation of one-dimensional monotonic utility functions based on experimental data. *Control & Cybernetics, 38*(3), 835-861.

Nikolova, N., Toneva-Zheynova, D., & Tenekedjiev, K. (2011). Classification Of Lotteries In Quantitative Decision Analysis. *Maritime Scientific Forum, Electronics, Electrotechniques and Automation Control. Informatics. Mathematics, 4*, 200-210.

Nikolova, N., Toneva, D., Ahmed, S., & Tenekedjiev, K. (2008b, May 15 -17). *Constructing multi-dimensional utility function under different types of preferences over the fundamental vector attributes.* Paper presented at the From Natural Language to Soft Computing: New Paradigms in Artificial Intelligence. Exploratory Workship on NL- Computation, Băile Felix, Oradea, Romania.

Nikolova, N. D., Dimitrakiev, D. J., & Tenekedjiev, K. I. (2004, 22-24 June 2004). *Fuzzy rationality in the elicitation of subjective probabilities.* Paper presented at the 2004 2nd International IEEE Conference on 'Intelligent Systems'. Proceedings (IEEE Cat. No.04EX791), Varna, Bulgaria.

Nikolova, N. D., Mednikarov, B., Tenekedjiev, K. . (2012, 3 - 7 October). *Analytical Version of the Non-Uniform Method for Analysing Uncertain Scaling Constants.* Paper presented at the International Conference Automatics and Informatics, Sofia, Bulgaria.

Ochiai, S., Makita, T., Sanjo, T., Sato, K., & Katagiri, M. (2005). Quantitative risk evaluations of LNG equipment applying ASME risk ‐ based maintenance concepts. *Process safety progress, 24*(3), 187-191.

Papakostas, N., Papachatzakis, P., Xanthakis, V., Mourtzis, D., & Chryssolouris, G. (2010). An approach to operational aircraft maintenance planning. *Decision Support Systems, 48*(4), 604-612.

Pariazar, M., Shahrabi, J., Zaeri, M., & Parhizi, S. (2008). A combined approach for maintenance strategy selection. *Journal of Applied Sciences, 8*(23), 4321-4329.

Paté-Cornell, E., & Fischbeck, P. S. (1993). PRA as a management tool: organizational factors and risk-based priorities for the maintenance of the tiles of the space shuttle orbiter. *Reliability Engineering & System Safety, 40*(3), 239-257.

Pratt, J. W., Raiffa, H., & Schlaifer, R. (2008). *Introduction to Statistical Decision Theory* (3 ed. Vol. 1). Massachusetts Institute of Technology, Cambridge, Massachusetts: MIT Press.

Rapoport, A. (1989). *Decision Theory and Decision Behaviour: Normative and Descriptive Approaches* (1 ed.). Heidelberg, Germany: Springer Netherlands.

Renooij, S. (2001). Probability elicitation for belief networks: issues to consider. *The Knowledge Engineering Review, 16*(3), 255-269.

Ries, J., González-Ramírez, R. G., & Voß, S. (2017). *Review of Fuzzy Techniques in Maritime Shipping Operations.* Paper presented at the Computational Logistics: 8th International Conference, ICCL 2017, Southampton, England.

Saaty, T. L. (2012). *Decision Making for Leaders: The Analytic Hierarchy Process for Decisions in a Complex World - New Edition* (3rd Revised Edition ed.). Pittsburg, Pennsylvania, United States of America: RWS Publications.

Saaty, T. L., & Vargas, L. G. (2012). *Models, Methods, Concepts & Applications of the Analytic Hierarchy Process* (2 ed. Vol. 175). New York, USA: Springer Science and Business Media.

Sabatino, S. M. (2017). *Probabilistic optimal decision making and life-cycle management considering risk, sustainability, and utility: applications to bridges and ships.* (Doctor of Philosophy in Structural Engineering), Lehigh University.

Sakthivel, N., Sugumaran, V., & Babudevasenapati, S. (2010). Vibration based fault diagnosis of monoblock centrifugal pump using decision tree. *Expert Systems with Applications, 37*(6), 4040-4049.

Sarkar, A., & Behera, D. K. (2012). Design of Optimal Maintenance Strategy. In S. Sathiyamoorthy, B. E. Caroline & J. G. Jayanthi (Eds.), *Emerging Trends in Science, Engineering and Technology: Proceedings of International Conference, INCOSET 2012* (pp. 383-389). India: Springer India.

Sawalhi, N., Randall, R. B., & Endo, H. (2007). The enhancement of fault detection and diagnosis in rolling element bearings using minimum entropy deconvolution combined with spectral kurtosis. *Mechanical Systems and Signal Processing, 21*(6), 2616-2633.

Schulz, V., & Stehfest, H. (1984). Regional energy supply optimization with multiple objectives. *European Journal of Operational Research, 17*(3), 302-312.

Sebastiani, L., Pescetto, A., & Ambrosio, L. (2013). *The Condition Monitoring System for Optimal Maintenance – Possible Application on Offshore Vessels*. Paper presented at the Offshore Mediterranean Conference, Ravenna, Italy.

Shahin, A., Shirouyehzad, H., & Pourjavad, E. (2012). Optimum maintenance strategy: a case study in the mining industry. *International Journal of Services and Operations Management, 12*(3), 368-386.

Shorten, D. C. (2013). *Marine machinery condition monitoring: Why has the shipping industry been slow to adopt?* Paper presented at the 10th International Conference on Condition Monitoring, CM 2013 and Machinery Failure Prevention Technologies 2013, MFPT 2013, June 18, 2013 - June 20, 2013, Krakow, Poland.

Smith, B. D. (1989). Risk Management in Repair Work Decision Making. *Naval engineers journal, 101*(3), 220-230.

Soliman, M., Barone, G., & Frangopol, D. M. (2015). Fatigue reliability and service life prediction of aluminum naval ship details based on monitoring data. *Structural Health Monitoring, 14*(1), 3-19.

Steadman, B., Berghout, F., Olsen, N., & Sorensen, B. (2008). *Intermittent fault detection and isolation system.* Paper presented at the 2008 IEEE AUTOTESTCON, Salt Lake City, Utah, USA.

Stephens, C. R., Huerta, H. F., & Linares, A. R. (2015, 6-8 July 2015). *Why the Naive Bayes approximation is not as Naive as it appears.* Paper presented at the 2015 6th International Conference on Information, Intelligence, Systems and Applications (IISA), Corfu, Greece.

Sun, J., Chen, D., Li, C., & Yan, H. (2018). Integration of scheduled structural health monitoring with airline maintenance program based on risk analysis. *Proceedings of the Institution of Mechanical Engineers, Part O: Journal of Risk and Reliability, 232*(1), 92-104.

Teixeira de Almeida, A. (2001). Repair contract decision model through additive utility function. *Journal of Quality in Maintenance Engineering, 7*(1), 42-48.

Tenekedjiev, K. (2007a). Triple bisection algorithms for 1-D utility elicitation in the case of monotonic preferences. *Advanced Studies in Contemporary Mathematics, 14*(2), 259-281.

Tenekedjiev, K., Kobashikawa, C. A., Nikolova, N. D., & Hirota, K. (2006). Generic Database for Hybrid Bayesian Pattern Recognition. *Journal of Advanced Computational Intelligence and Intelligent Informatics, 10*(3), 419-431.

Tenekedjiev, K. I. (2007b). Uncertain equivalence method for continuous one-dimensional utility elicitation. *International Federation of Automatic Control Proceedings Volumes, 40*(18), 195-202.

Tenekedjiev, K. I., & Nikolova, N. D. (2008). Ranking discrete outcome alternatives with partially quantified uncertainty. *International journal of general systems, 37*(2), 249-274.

The MathWorks Inc. (2018). MATLAB (Version R2018b).

Ting, K. M. (2017). Confusion Matrix. In C. Sammut & G. I. Webb (Eds.), *Encyclopedia of Machine Learning and Data Mining* (2 ed., pp. 260-260). Boston, MA: Springer US.

Triantaphyllou, E. (2013). *Multi-criteria Decision Making Methods: A Comparative Study*. Boston, MA: Springer Science & Business Media.

Turgut, S., Dağtekin, M., & Ensari, T. (2018). *Microarray breast cancer data classification using machine learning methods.* Paper presented at the 2018 Electric Electronics, Computer Science, Biomedical Engineerings' Meeting (EBBT), Istanbul, Turkey.

Ulusoy, G., Or, I., & Soydan, N. (1992). Design and implementation of a maintenance planning and control system. *International Journal of Production Economics, 24*(3), 263-272.

United States Air Force. (2019). Fiscal Year 2020 Budget Overview.

Von Neumann, J., & Morgenstern, O. (1947). *Theory of Games and Economic Behavior, 2nd Revised Edition* (2 ed.). Princeton, New Jersey, United States of America: Princeton Univeristy Press.

Wahid, A.-S. A., Ahmad, M. Z., Ahmad, K. A., Taylor, J., Abdullah, A., Al-Shafiq, A.-A., . . . Kitagawa, K. (2018a). Demystifying Ship Operational Availability: An Innovative Approach for Management of In-Service Support Contracts. *Science and Technology Research Institute for Defence (STRIDE), 181*, 338.

Wahid, A.-S. B. A., Zamani, M., Sunarsih, M. N. B. A. G., Yolhamid, M., Zarim, A. U. A. A., Abdullah, A. B., & Azlan, N. H. A. (2018b). Severity of Downtime Influence Factors Impacting Naval Ship Operational Availability - A Five-Stage Delphi Consensus Procedure with Snowballing. *Asian Research Publishing Network (APRN) Journal of Engineering and Applied Sicences, 13*(3), 939-946.

Weinberger, K. Q., Blitzer, J., & Saul, L. K. (2009). Distance Metric Learning for Large Margin Nearest Neighbor Classification. *Journal of Machine Learning Research, 10*(Feb), 207-244.

Williams, C. M., & Hester, P. T. (2017). A Readiness Decision Model for Canceling Navy Ship Maintenance Availabilities. In K. D. Lawrence & G. Kleinman (Eds.), *Applications of Management Science* (Vol. 18, pp. 147-166). Bingley, UK: Emerald Group Publishing.

Woitek, R., Spick, C., Schernthaner, M., Rudas, M., Kapetas, P., Bernathova, M., . . . Baltzer, P. A. T. (2017). A simple classification system (the Tree flowchart) for breast MRI can reduce the number of unnecessary biopsies in MRI-only lesions. *European Radiology, 27*(9), 3799-3809.

Xu, A. Y., Bhatnagar, J., Bednarz, G., Flickinger, J., Arai, Y., Vacsulka, J., . . . Lunsford, L. D. (2017). Failure modes and effects analysis (FMEA) for Gamma Knife radiosurgery. *Journal of applied clinical medical physics, 18*(6), 152-168.

Yatomi, M., Fuji, A., Saito, N., & Yoshida, T. (2004). *An Approach for Cost Effective Assessment in Risk-Based Maintenance as a Life-Cycle Maintenance (LCM) Model.* Paper presented at the ASME/JSME 2004 Pressure Vessels and Piping Conference, San Diego, California, USA.

Yıldızbaşı, A., & Daneshvar Rouyendegh, B. (2018). Multi-Criteria Decision Making Approach for Evaluation of the Performance of Computer Programming Languages in Higher Education. *Computer Applications in Engineering Education, 26*(6), 1992-2001.

Zhang, H. (2005). Exploring conditions for the optimality of naive Bayes. *International Journal of Pattern Recognition and Artificial Intelligence, 19*(02), 183-198.

APPENDIX A - Derivation of Posterior Probability Formulae

The present Appendix A presents the derivation of equations (2-16) and (2-17) described in Chapter 2, beginning with (2-11).

From Chapter 2, a general Bayesian linear discriminant function $g_k(\vec{x})$ where $\vec{x}$ is some vector of measurements can be described by (A-1) as follows (reproduced from 2-11):

$$g_k(\vec{x}) = \ln p(\vec{x}\,|c_k) + \ln P(c_k) \tag{A-1}$$

Because the discriminant functions are Bayesian, *let*:

$$g_k(\vec{x}) = P(c_k\,|\,\vec{x})$$

Rearranging:

$$\ln p(\vec{x}\,|c_k) = g_k(\vec{x}) - \ln P(c_k)$$

Taking the exponentials of both sides and letting $A = \ln P(c_k)$, the general form (A-2) can be derived:

$$p(\vec{x}\,|c_k) = e^{g_k(\vec{x})} \times e^{A} \tag{A-2}$$

Accordingly, the summation across all classes c_k is:

$$\sum_{k=1}^{c} p(\vec{x}\,|\,c_k) = 1 = e^{A} \times \sum_{k=1}^{c} e^{g_k(\vec{x})}$$

And this can be rearranged in terms of e^A to obtain a new expression (A-3):

$$e^{A} = \frac{1}{\sum_{k=1}^{c} e^{g_k(\vec{x})}} \tag{A-3}$$

Substituting (A-3) into (A-2), yields (A-4) (or (2-17) in Chapter 2):

$$P(c_k \mid \vec{x}) = \frac{e^{\,g_k(\vec{x})}}{\sum_{i=1}^{k} e^{\,g_k(\vec{x})}} \tag{A-4}$$

Lastly, taking the inverse and applying exponential rules for the division of two terms, gives an equivalent representation shown in (A-5) (or (2-16) in Chapter 2), whose implementation reduces computational time in comparison as only one exponential term must be calculated.

$$P(c_k \mid \vec{x}) = \frac{1}{\sum_{i=1}^{k} e^{\left(\, g_i(\vec{x}) - g_k(\vec{x}) \,\right)}} \tag{A-5}$$

APPENDIX B - Number 2 General Service Pump Maintenance Record

The present Appendix B presents the complied maintenance record for the Number 2 General Service pump, between 01/03/2015 to 05/08/2018.

NO. 2 GENERAL SERVICE PUMP MAINTENANCE RECORD
BEGINNING **1 MARCH 2015** TO END **29 JULY 2018**

DATE	MAINTENANCE TYPE	TASK DESCRIPTION	TASK DETAILS	NOTES
1/03/2015	Preventative Maintenance	Check drive-end bearing grease	Check drive-end bearing grease and add grease as required	No additional grease added
1/03/2015	Preventative Maintenance	Change drive-end bearing grease	Change drive-end bearing grease	Cleaned and re-greased
1/03/2015	Preventative Maintenance	Change non-drive-end sleeve grease	Check non-drive-end sleeve	Cleaned and lightly greased
1/03/2015	Corrective Maintenance	Replace packing in pump	Replaced graphite packing with more suitable teflon packing. Packing was also found to be initially too tight upon inspection. Also loosened packing. Further, change of operational mode was required to allow for sufficient lubrication of packing - some back pressure applied.	Replaced part (packing)
1/03/2015	Corrective Maintenance	Realign pump	Removed pump from base (ref. 'Repaint pump base'). Upon replacement of pump and motor, added shims to align pump and motor.	Realigned
1/03/2015	Corrective Maintenance	Repaint pump base	Strip and repaint pump base to minimise current and future effects of corrosion	Base repainted
1/03/2015	Preventative Maintenance	General clean of pump and motor	Remove dust, debris from pump and motor	Completed
1/03/2015	Preventative Maintenance	Visual inspection of pump and motor	Perform visual inspection of pump and motor	Completed
8/03/2015	Preventative Maintenance	Visual inspection of pump and motor	Perform visual inspection of pump and motor	Completed
15/03/2015	Preventative Maintenance	Visual inspection of pump and motor	Perform visual inspection of pump and motor	Completed
22/03/2015	Preventative Maintenance	Visual inspection of pump and motor	Perform visual inspection of pump and motor	Completed
29/03/2015	Preventative Maintenance	Visual inspection of pump and motor	Perform visual inspection of pump and motor	Completed
1/04/2015	Preventative Maintenance	Check drive-end bearing grease	Check drive-end bearing grease and add grease as required	No additional grease added
8/04/2015	Preventative Maintenance	Visual inspection of pump and motor	Perform visual inspection of pump and motor	Completed
15/04/2015	Preventative Maintenance	Visual inspection of pump and motor	Perform visual inspection of pump and motor	Completed
22/04/2015	Preventative Maintenance	Visual inspection of pump and motor	Perform visual inspection of pump and motor	Completed
29/04/2015	Preventative Maintenance	Visual inspection of pump and motor	Perform visual inspection of pump and motor	Completed
1/05/2015	Preventative Maintenance	Check drive-end bearing grease	Check drive-end bearing grease and add grease as required	No additional grease added
8/05/2015	Preventative Maintenance	Visual inspection of pump and motor	Perform visual inspection of pump and motor	Completed
15/05/2015	Preventative Maintenance	Visual inspection of pump and motor	Perform visual inspection of pump and motor	Completed
22/05/2015	Preventative Maintenance	Visual inspection of pump and motor	Perform visual inspection of pump and motor	Completed
29/05/2015	Preventative Maintenance	Visual inspection of pump and motor	Perform visual inspection of pump and motor	Completed
1/06/2015	Preventative Maintenance	Check drive-end bearing grease	Check drive-end bearing grease and add grease as required	2g Fresh grease added
5/06/2015	Preventative Maintenance	Visual inspection of pump and motor	Perform visual inspection of pump and motor	Completed
8/06/2015	Preventative Maintenance	Visual inspection of pump and motor	Perform visual inspection of pump and motor	Completed
15/06/2015	Preventative Maintenance	Visual inspection of pump and motor	Perform visual inspection of pump and motor	Completed
22/06/2015	Preventative Maintenance	Visual inspection of pump and motor	Perform visual inspection of pump and motor	Completed
29/06/2015	Preventative Maintenance	Visual inspection of pump and motor	Perform visual inspection of pump and motor	Completed
1/07/2015	Preventative Maintenance	Check drive-end bearing grease	Check drive-end bearing grease and add grease as required	No additional grease added
8/07/2015	Preventative Maintenance	Visual inspection of pump and motor	Perform visual inspection of pump and motor	Completed
15/07/2015	Preventative Maintenance	Visual inspection of pump and motor	Perform visual inspection of pump and motor	Completed
22/07/2015	Preventative Maintenance	Visual inspection of pump and motor	Perform visual inspection of pump and motor	Completed
29/07/2015	Preventative Maintenance	Visual inspection of pump and motor	Perform visual inspection of pump and motor	Completed
1/08/2015	Preventative Maintenance	Check drive-end bearing grease	Check drive-end bearing grease and add grease as required	No additional grease added
8/08/2015	Preventative Maintenance	Visual inspection of pump and motor	Perform visual inspection of pump and motor	Completed
15/08/2015	Preventative Maintenance	Visual inspection of pump and motor	Perform visual inspection of pump and motor	Completed
22/08/2015	Preventative Maintenance	Visual inspection of pump and motor	Perform visual inspection of pump and motor	Completed
29/08/2015	Preventative Maintenance	Visual inspection of pump and motor	Perform visual inspection of pump and motor	Completed
1/09/2015	Preventative Maintenance	Check drive-end bearing grease	Check drive-end bearing grease and add grease as required	2g Fresh grease added
5/09/2015	Preventative Maintenance	Visual inspection of pump and motor	Perform visual inspection of pump and motor	Completed
8/09/2015	Preventative Maintenance	Visual inspection of pump and motor	Perform visual inspection of pump and motor	Completed
15/09/2015	Preventative Maintenance	Visual inspection of pump and motor	Perform visual inspection of pump and motor	Completed
22/09/2015	Preventative Maintenance	Visual inspection of pump and motor	Perform visual inspection of pump and motor	Completed
29/09/2015	Preventative Maintenance	Visual inspection of pump and motor	Perform visual inspection of pump and motor	Completed
1/10/2015	Preventative Maintenance	Check drive-end bearing grease	Check drive-end bearing grease and add grease as required	No additional grease added
8/10/2015	Preventative Maintenance	Visual inspection of pump and motor	Perform visual inspection of pump and motor	Completed
15/10/2015	Preventative Maintenance	Visual inspection of pump and motor	Perform visual inspection of pump and motor	Completed
22/10/2015	Preventative Maintenance	Visual inspection of pump and motor	Perform visual inspection of pump and motor	Completed
29/10/2015	Preventative Maintenance	Visual inspection of pump and motor	Perform visual inspection of pump and motor	Completed
1/11/2015	Preventative Maintenance	Check drive-end bearing grease	Check drive-end bearing grease and add grease as required	No additional grease added
8/11/2015	Preventative Maintenance	Visual inspection of pump and motor	Perform visual inspection of pump and motor	Completed
15/11/2015	Preventative Maintenance	Visual inspection of pump and motor	Perform visual inspection of pump and motor	Completed
22/11/2015	Preventative Maintenance	Visual inspection of pump and motor	Perform visual inspection of pump and motor	Completed
29/11/2015	Preventative Maintenance	Visual inspection of pump and motor	Perform visual inspection of pump and motor	Completed
1/12/2015	Preventative Maintenance	Check drive-end bearing grease	Check drive-end bearing grease and add grease as required	2g Fresh grease added
6/12/2015	Preventative Maintenance	Visual inspection of pump and motor	Perform visual inspection of pump and motor	Completed
8/12/2015	Preventative Maintenance	Visual inspection of pump and motor	Perform visual inspection of pump and motor	Completed
15/12/2015	Preventative Maintenance	Visual inspection of pump and motor	Perform visual inspection of pump and motor	Completed
22/12/2015	Preventative Maintenance	Visual inspection of pump and motor	Perform visual inspection of pump and motor	Completed
29/12/2015	Preventative Maintenance	Visual inspection of pump and motor	Perform visual inspection of pump and motor	Completed
1/01/2016	Preventative Maintenance	Check drive-end bearing grease	Check drive-end bearing grease and add grease as required	No additional grease added
8/01/2016	Preventative Maintenance	Visual inspection of pump and motor	Perform visual inspection of pump and motor	Completed
15/01/2016	Preventative Maintenance	Visual inspection of pump and motor	Perform visual inspection of pump and motor	Completed
22/01/2016	Preventative Maintenance	Visual inspection of pump and motor	Perform visual inspection of pump and motor	Completed
29/01/2016	Preventative Maintenance	Visual inspection of pump and motor	Perform visual inspection of pump and motor	Completed
1/02/2016	Preventative Maintenance	Check drive-end bearing grease	Check drive-end bearing grease and add grease as required	No additional grease added
5/02/2016	Preventative Maintenance	Visual inspection of pump and motor	Perform visual inspection of pump and motor	Completed
8/02/2016	Preventative Maintenance	Visual inspection of pump and motor	Perform visual inspection of pump and motor	Completed
15/02/2016	Preventative Maintenance	Visual inspection of pump and motor	Perform visual inspection of pump and motor	Completed
22/02/2016	Preventative Maintenance	Visual inspection of pump and motor	Perform visual inspection of pump and motor	Completed
29/02/2016	Preventative Maintenance	Visual inspection of pump and motor	Perform visual inspection of pump and motor	Completed
1/03/2016	Preventative Maintenance	Check drive-end bearing grease	Check drive-end bearing grease and add grease as required	2g Fresh grease added
1/03/2016	Preventative Maintenance	General clean of pump and motor	Remove dust, debris from pump and motor	Completed
8/03/2016	Preventative Maintenance	Visual inspection of pump and motor	Perform visual inspection of pump and motor	Completed
15/03/2016	Preventative Maintenance	Visual inspection of pump and motor	Perform visual inspection of pump and motor	Completed
22/03/2016	Preventative Maintenance	Visual inspection of pump and motor	Perform visual inspection of pump and motor	Completed
29/03/2016	Preventative Maintenance	Visual inspection of pump and motor	Perform visual inspection of pump and motor	Completed
1/04/2016	Preventative Maintenance	Check drive-end bearing grease	Check drive-end bearing grease and add grease as required	No additional grease added
8/04/2016	Preventative Maintenance	Visual inspection of pump and motor	Perform visual inspection of pump and motor	Completed
15/04/2016	Preventative Maintenance	Visual inspection of pump and motor	Perform visual inspection of pump and motor	Completed
22/04/2016	Preventative Maintenance	Visual inspection of pump and motor	Perform visual inspection of pump and motor	Completed
29/04/2016	Preventative Maintenance	Visual inspection of pump and motor	Perform visual inspection of pump and motor	Completed
1/05/2016	Preventative Maintenance	Check drive-end bearing grease	Check drive-end bearing grease and add grease as required	No additional grease added

DATE	MAINTENANCE TYPE	TASK DESCRIPTION	TASK DETAILS	NOTES
8/05/2016	Preventative Maintenance	Visual inspection of pump and motor	Perform visual inspection of pump and motor	Completed
15/05/2016	Preventative Maintenance	Visual inspection of pump and motor	Perform visual inspection of pump and motor	Completed
22/05/2016	Preventative Maintenance	Visual inspection of pump and motor	Perform visual inspection of pump and motor	Completed
29/05/2016	Preventative Maintenance	Visual inspection of pump and motor	Perform visual inspection of pump and motor	Completed
1/06/2016	Preventative Maintenance	Check drive-end bearing grease	Check drive-end bearing grease and add grease as required	2g Fresh grease added
5/06/2016	Preventative Maintenance	Visual inspection of pump and motor	Perform visual inspection of pump and motor	Completed
8/06/2016	Preventative Maintenance	Visual inspection of pump and motor	Perform visual inspection of pump and motor	Completed
15/06/2016	Preventative Maintenance	Visual inspection of pump and motor	Perform visual inspection of pump and motor	Completed
22/06/2016	Preventative Maintenance	Visual inspection of pump and motor	Perform visual inspection of pump and motor	Completed
29/06/2016	Preventative Maintenance	Visual inspection of pump and motor	Perform visual inspection of pump and motor	Completed
1/07/2016	Preventative Maintenance	Check drive-end bearing grease	Check drive-end bearing grease and add grease as required	No additional grease added
8/07/2016	Preventative Maintenance	Visual inspection of pump and motor	Perform visual inspection of pump and motor	Completed
15/07/2016	Preventative Maintenance	Visual inspection of pump and motor	Perform visual inspection of pump and motor	Completed
22/07/2016	Preventative Maintenance	Visual inspection of pump and motor	Perform visual inspection of pump and motor	Completed
29/07/2016	Preventative Maintenance	Visual inspection of pump and motor	Perform visual inspection of pump and motor	Completed
1/08/2016	Preventative Maintenance	Check drive-end bearing grease	Check drive-end bearing grease and add grease as required	No additional grease added
5/08/2016	Preventative Maintenance	Visual inspection of pump and motor	Perform visual inspection of pump and motor	Completed
8/08/2016	Preventative Maintenance	Visual inspection of pump and motor	Perform visual inspection of pump and motor	Completed
15/08/2016	Preventative Maintenance	Visual inspection of pump and motor	Perform visual inspection of pump and motor	Completed
22/08/2016	Preventative Maintenance	Visual inspection of pump and motor	Perform visual inspection of pump and motor	Completed
29/08/2016	Preventative Maintenance	Visual inspection of pump and motor	Perform visual inspection of pump and motor	Completed
1/09/2016	Preventative Maintenance	Check drive-end bearing grease	Check drive-end bearing grease and add grease as required	2g Fresh grease added
8/09/2016	Preventative Maintenance	Visual inspection of pump and motor	Perform visual inspection of pump and motor	Completed
15/09/2016	Preventative Maintenance	Visual inspection of pump and motor	Perform visual inspection of pump and motor	Completed
22/09/2016	Preventative Maintenance	Visual inspection of pump and motor	Perform visual inspection of pump and motor	Completed
29/09/2016	Preventative Maintenance	Visual inspection of pump and motor	Perform visual inspection of pump and motor	Completed
1/10/2016	Preventative Maintenance	Check drive-end bearing grease	Check drive-end bearing grease and add grease as required	No additional grease added
8/10/2016	Preventative Maintenance	Visual inspection of pump and motor	Perform visual inspection of pump and motor	Completed
15/10/2016	Preventative Maintenance	Visual inspection of pump and motor	Perform visual inspection of pump and motor	Completed
22/10/2016	Preventative Maintenance	Visual inspection of pump and motor	Perform visual inspection of pump and motor	Completed
29/10/2016	Preventative Maintenance	Visual inspection of pump and motor	Perform visual inspection of pump and motor	Completed
1/11/2016	Preventative Maintenance	Check drive-end bearing grease	Check drive-end bearing grease and add grease as required	No additional grease added
5/11/2016	Preventative Maintenance	Visual inspection of pump and motor	Perform visual inspection of pump and motor	Completed
8/11/2016	Preventative Maintenance	Visual inspection of pump and motor	Perform visual inspection of pump and motor	Completed
15/11/2016	Preventative Maintenance	Visual inspection of pump and motor	Perform visual inspection of pump and motor	Completed
22/11/2016	Preventative Maintenance	Visual inspection of pump and motor	Perform visual inspection of pump and motor	Completed
29/11/2016	Preventative Maintenance	Visual inspection of pump and motor	Perform visual inspection of pump and motor	Completed
1/12/2016	Preventative Maintenance	Check drive-end bearing grease	Check drive-end bearing grease and add grease as required	2g Fresh grease added
8/12/2016	Preventative Maintenance	Visual inspection of pump and motor	Perform visual inspection of pump and motor	Completed
15/12/2016	Preventative Maintenance	Visual inspection of pump and motor	Perform visual inspection of pump and motor	Completed
22/12/2016	Preventative Maintenance	Visual inspection of pump and motor	Perform visual inspection of pump and motor	Completed
29/12/2016	Preventative Maintenance	Visual inspection of pump and motor	Perform visual inspection of pump and motor	Completed
1/01/2017	Preventative Maintenance	Check drive-end bearing grease	Check drive-end bearing grease and add grease as required	No additional grease added
8/01/2017	Preventative Maintenance	Visual inspection of pump and motor	Perform visual inspection of pump and motor	Completed
15/01/2017	Preventative Maintenance	Visual inspection of pump and motor	Perform visual inspection of pump and motor	Completed
22/01/2017	Preventative Maintenance	Visual inspection of pump and motor	Perform visual inspection of pump and motor	Completed
29/01/2017	Preventative Maintenance	Visual inspection of pump and motor	Perform visual inspection of pump and motor	Completed
1/02/2017	Preventative Maintenance	Check drive-end bearing grease	Check drive-end bearing grease and add grease as required	No additional grease added
5/02/2017	Preventative Maintenance	Visual inspection of pump and motor	Perform visual inspection of pump and motor	Completed
8/02/2017	Preventative Maintenance	Visual inspection of pump and motor	Perform visual inspection of pump and motor	Completed
15/02/2017	Preventative Maintenance	Visual inspection of pump and motor	Perform visual inspection of pump and motor	Completed
22/02/2017	Preventative Maintenance	Visual inspection of pump and motor	Perform visual inspection of pump and motor	Completed
1/03/2017	Preventative Maintenance	Visual inspection of pump and motor	Perform visual inspection of pump and motor	Completed
1/03/2017	Preventative Maintenance	Change drive-end bearing grease	Change drive-end bearing grease	Cleaned and re-greased
1/03/2017	Preventative Maintenance	Change non-drive-end sleeve grease	Change non-drive-end sleeve grease	Cleaned and lightly greased
1/03/2017	Preventative Maintenance	Function test after maintenance	Perform function test of pump and monitor bearing and sleeve	Completed
1/03/2017	Preventative Maintenance	General clean of pump and motor	Remove dust, debris from pump and motor	Completed
8/03/2017	Preventative Maintenance	Visual inspection of pump and motor	Perform visual inspection of pump and motor	Completed
15/03/2017	Preventative Maintenance	Visual inspection of pump and motor	Perform visual inspection of pump and motor	Completed
22/03/2017	Preventative Maintenance	Visual inspection of pump and motor	Perform visual inspection of pump and motor	Completed
29/03/2017	Preventative Maintenance	Visual inspection of pump and motor	Perform visual inspection of pump and motor	Completed
1/04/2017	Preventative Maintenance	Check drive-end bearing grease	Check drive-end bearing grease and add grease as required	No additional grease added
8/04/2017	Preventative Maintenance	Visual inspection of pump and motor	Perform visual inspection of pump and motor	Completed
15/04/2017	Preventative Maintenance	Visual inspection of pump and motor	Perform visual inspection of pump and motor	Completed
22/04/2017	Preventative Maintenance	Visual inspection of pump and motor	Perform visual inspection of pump and motor	Completed
29/04/2017	Preventative Maintenance	Visual inspection of pump and motor	Perform visual inspection of pump and motor	Completed
1/05/2017	Preventative Maintenance	Check drive-end bearing grease	Check drive-end bearing grease and add grease as required	No additional grease added
6/05/2017	Preventative Maintenance	Visual inspection of pump and motor	Perform visual inspection of pump and motor	Completed
8/05/2017	Preventative Maintenance	Visual inspection of pump and motor	Perform visual inspection of pump and motor	Completed
15/05/2017	Preventative Maintenance	Visual inspection of pump and motor	Perform visual inspection of pump and motor	Completed
22/05/2017	Preventative Maintenance	Visual inspection of pump and motor	Perform visual inspection of pump and motor	Completed
29/05/2017	Preventative Maintenance	Visual inspection of pump and motor	Perform visual inspection of pump and motor	Completed
1/06/2017	Preventative Maintenance	Check drive-end bearing grease	Check drive-end bearing grease and add grease as required	2g Fresh grease added
8/06/2017	Preventative Maintenance	Visual inspection of pump and motor	Perform visual inspection of pump and motor	Completed
15/06/2017	Preventative Maintenance	Visual inspection of pump and motor	Perform visual inspection of pump and motor	Completed
22/06/2017	Preventative Maintenance	Visual inspection of pump and motor	Perform visual inspection of pump and motor	Completed
29/06/2017	Preventative Maintenance	Visual inspection of pump and motor	Perform visual inspection of pump and motor	Completed
1/07/2017	Preventative Maintenance	Check drive-end bearing grease	Check drive-end bearing grease and add grease as required	No additional grease added
8/07/2017	Preventative Maintenance	Visual inspection of pump and motor	Perform visual inspection of pump and motor	Completed
15/07/2017	Preventative Maintenance	Visual inspection of pump and motor	Perform visual inspection of pump and motor	Completed
22/07/2017	Preventative Maintenance	Visual inspection of pump and motor	Perform visual inspection of pump and motor	Completed
29/07/2017	Preventative Maintenance	Visual inspection of pump and motor	Perform visual inspection of pump and motor	Completed
1/08/2017	Preventative Maintenance	Check drive-end bearing grease	Check drive-end bearing grease and add grease as required	No additional grease added
5/08/2017	Preventative Maintenance	Visual inspection of pump and motor	Perform visual inspection of pump and motor	Completed
8/08/2017	Preventative Maintenance	Visual inspection of pump and motor	Perform visual inspection of pump and motor	Completed
15/08/2017	Preventative Maintenance	Visual inspection of pump and motor	Perform visual inspection of pump and motor	Completed
22/08/2017	Preventative Maintenance	Visual inspection of pump and motor	Perform visual inspection of pump and motor	Completed
29/08/2017	Preventative Maintenance	Visual inspection of pump and motor	Perform visual inspection of pump and motor	Completed
1/09/2017	Preventative Maintenance	Check drive-end bearing grease	Check drive-end bearing grease and add grease as required	2g Fresh grease added
8/09/2017	Preventative Maintenance	Visual inspection of pump and motor	Perform visual inspection of pump and motor	Completed
15/09/2017	Preventative Maintenance	Visual inspection of pump and motor	Perform visual inspection of pump and motor	Completed
22/09/2017	Preventative Maintenance	Visual inspection of pump and motor	Perform visual inspection of pump and motor	Completed
29/09/2017	Preventative Maintenance	Visual inspection of pump and motor	Perform visual inspection of pump and motor	Completed
1/10/2017	Preventative Maintenance	Check drive-end bearing grease	Check drive-end bearing grease and add grease as required	No additional grease added
8/10/2017	Preventative Maintenance	Visual inspection of pump and motor	Perform visual inspection of pump and motor	Completed
15/10/2017	Preventative Maintenance	Visual inspection of pump and motor	Perform visual inspection of pump and motor	Completed
22/10/2017	Preventative Maintenance	Visual inspection of pump and motor	Perform visual inspection of pump and motor	Completed
29/10/2017	Preventative Maintenance	Visual inspection of pump and motor	Perform visual inspection of pump and motor	Completed
1/11/2017	Preventative Maintenance	Check drive-end bearing grease	Check drive-end bearing grease and add grease as required	No additional grease added

DATE	MAINTENANCE TYPE	TASK DESCRIPTION	TASK DETAILS	NOTES	
5/11/2017	Preventative Maintenance	Visual inspection of pump and motor	Perform visual inspection of pump and motor	Completed	
8/11/2017	Preventative Maintenance	Visual inspection of pump and motor	Perform visual inspection of pump and motor	Completed	
15/11/2017	Preventative Maintenance	Visual inspection of pump and motor	Perform visual inspection of pump and motor	Completed	
22/11/2017	Preventative Maintenance	Visual inspection of pump and motor	Perform visual inspection of pump and motor	Completed	
29/11/2017	Preventative Maintenance	Visual inspection of pump and motor	Perform visual inspection of pump and motor	Completed	
1/12/2017	Preventative Maintenance	Check drive-end bearing grease	Check drive-end bearing grease and add grease as required	2g Fresh grease added	
8/12/2017	Preventative Maintenance	Visual inspection of pump and motor	Perform visual inspection of pump and motor	Completed	
15/12/2017	Preventative Maintenance	Visual inspection of pump and motor	Perform visual inspection of pump and motor	Completed	
22/12/2017	Preventative Maintenance	Visual inspection of pump and motor	Perform visual inspection of pump and motor	Completed	
29/12/2017	Preventative Maintenance	Visual inspection of pump and motor	Perform visual inspection of pump and motor	Completed	
1/01/2018	Preventative Maintenance	Check drive-end bearing grease	Check drive-end bearing grease and add grease as required	No additional grease added	
5/01/2018	Preventative Maintenance	Visual inspection of pump and motor	Perform visual inspection of pump and motor	Completed	
8/01/2018	Preventative Maintenance	Visual inspection of pump and motor	Perform visual inspection of pump and motor	Completed	
15/01/2018	Preventative Maintenance	Visual inspection of pump and motor	Perform visual inspection of pump and motor	Completed	
22/01/2018	Preventative Maintenance	Visual inspection of pump and motor	Perform visual inspection of pump and motor	Completed	
29/01/2018	Preventative Maintenance	Visual inspection of pump and motor	Perform visual inspection of pump and motor	Completed	
1/02/2018	Preventative Maintenance	Check drive-end bearing grease	Check drive-end bearing grease and add grease as required	No additional grease added	
8/02/2018	Preventative Maintenance	Visual inspection of pump and motor	Perform visual inspection of pump and motor	Completed	
15/02/2018	Preventative Maintenance	Visual inspection of pump and motor	Perform visual inspection of pump and motor	Completed	
22/02/2018	Preventative Maintenance	Visual inspection of pump and motor	Perform visual inspection of pump and motor	Completed	
1/03/2018	Preventative Maintenance	Visual inspection of pump and motor	Perform visual inspection of pump and motor	Completed	
1/03/2018	Preventative Maintenance	Check drive-end bearing grease	Check drive-end bearing grease and add grease as required	2g Fresh grease added	
1/03/2018	Preventative Maintenance	General clean of pump and motor	Remove dust, debris from pump and motor	Completed	
8/03/2018	Preventative Maintenance	Visual inspection of pump and motor	Perform visual inspection of pump and motor	Completed	CM#1
15/03/2018	Preventative Maintenance	Visual inspection of pump and motor	Perform visual inspection of pump and motor	Completed	
22/03/2018	Preventative Maintenance	Visual inspection of pump and motor	Perform visual inspection of pump and motor	Completed	
29/03/2018	Preventative Maintenance	Visual inspection of pump and motor	Perform visual inspection of pump and motor	Completed	CM#2
1/04/2018	Preventative Maintenance	Check drive-end bearing grease	Check drive-end bearing grease and add grease as required	No additional grease added	
8/04/2018	Preventative Maintenance	Visual inspection of pump and motor	Perform visual inspection of pump and motor	Completed	CM#3
15/04/2018	Preventative Maintenance	Visual inspection of pump and motor	Perform visual inspection of pump and motor	Completed	
22/04/2018	Preventative Maintenance	Visual inspection of pump and motor	Perform visual inspection of pump and motor	Completed	
29/04/2018	Preventative Maintenance	Visual inspection of pump and motor	Perform visual inspection of pump and motor	Completed	CM#4
1/05/2018	Preventative Maintenance	Check drive-end bearing grease	Check drive-end bearing grease and add grease as required	No additional grease added	
6/05/2018	Preventative Maintenance	Visual inspection of pump and motor	Perform visual inspection of pump and motor	Completed	CM#5
8/05/2018	Preventative Maintenance	Visual inspection of pump and motor	Perform visual inspection of pump and motor	Completed	
15/05/2018	Preventative Maintenance	Visual inspection of pump and motor	Perform visual inspection of pump and motor	Completed	CM#6
22/05/2018	Preventative Maintenance	Visual inspection of pump and motor	Perform visual inspection of pump and motor	Completed	
29/05/2018	Preventative Maintenance	Visual inspection of pump and motor	Perform visual inspection of pump and motor	Completed	CM#7
1/06/2018	Preventative Maintenance	Check drive-end bearing grease	Check drive-end bearing grease and add grease as required	2g of fresh grease added	
8/06/2018	Preventative Maintenance	Visual inspection of pump and motor	Perform visual inspection of pump and motor	Completed	
15/06/2018	Preventative Maintenance	Visual inspection of pump and motor	Perform visual inspection of pump and motor	Completed	CM#8
22/06/2018	Preventative Maintenance	Visual inspection of pump and motor	Perform visual inspection of pump and motor	Completed	
29/06/2018	Preventative Maintenance	Visual inspection of pump and motor	Perform visual inspection of pump and motor	Completed	CM#9
1/07/2018	Preventative Maintenance	Check drive-end bearing grease	Check drive-end bearing grease and add grease as required	Completed	
5/07/2018	**Condition-based Maintenance**	**Stop pump and tighten packing**	**Tighten packing to recitfy excessing packing drip rate**	**Completed**	
8/07/2018	Preventative Maintenance	Visual inspection of pump and motor	Perform visual inspection of pump and motor	Completed	CM#10
15/07/2018	Preventative Maintenance	Visual inspection of pump and motor	Perform visual inspection of pump and motor	Completed	
22/07/2018	Preventative Maintenance	Visual inspection of pump and motor	Perform visual inspection of pump and motor	Completed	CM#11
29/07/2018	Preventative Maintenance	Visual inspection of pump and motor	Perform visual inspection of pump and motor	Completed	
5/08/2018	Preventative Maintenance	Visual inspection of pump and motor	Perform visual inspection of pump and motor	Completed	

APPENDIX C - Data Processing and Maintenance System Software

The present Appendix C presents the software developed in the process of the work conducted within the present Thesis. Initially, the Appendix presents the original software developed for Data Processing, then authorship of non-original functions used within the present maintenance system software is acknowledged and a small discussion provided for a function which can be used beyond the scope of the work in the present Thesis. Lastly, the original software developed for the shipboard pump maintenance system application is presented.

All code was written in MATLAB R2018b in combination with the Statistics and Machine Learning Toolbox, © 1994-2019 The MathWorks, Inc. All software has been published for inclusion within the present Thesis using the inbuilt tools as part of MATLAB R2018b.

C.1 Data Processing Software

This section presents the original software used to extract features from the raw vibration data obtained from the shipboard pump described in Chapter 3.

C.1.1 Software

The present section contains only original functions developed in the process of the present Thesis. The functions are used only to process vibration data. Their operation and outputs are described in Chapter 3. Outputs are produced in both cases as an Excel spreadsheet however can be modified as required for MATLAB use in future systems. Two functions are listed below and presented in alphabetical order.

processFFT.m

processWfm.m

```
function []=processFFT()
% processFFT converts a velocity FFT dataset from an excel spreadsheet
 into ML features
% Data extraction ranges are calibrated to the experimental data -
 there is
% some error in the vibration analyser or tachometer. Refer to
 relevant
% Thesis / publications
%Outputs are saved in a spreadsheet file

% CALCULATIONS:
    % 1. IMPORT DATA
    filename = 'your_filename';
    outputSuffix = '_FFTParams';
    outputFilename = strcat(filename,outputSuffix);
    data = xlsread(filename,'A18:B817');
    Frequency = data(:,1);
    Velocity = data(:,2);
    shaft_rotation = 1484; %shaft rotation speed in RPM
    numBlades = 21;
    numBrngBalls = 8;
    % 2. PROCESS DATA
    %Find the BPF given some tolerances in frequency values (0.5%)
    BPF_val = (shaft_rotation*numBlades)/60; %Calculate the blade pass
 frequency
    BPF_upper = BPF_val+(0.005*BPF_val);
    BPF_lower = BPF_val-(0.005*BPF_val);
    BPF_range = [];
    for i = 1:length(Frequency)
        if (Frequency(i) >= BPF_lower) && (BPF_upper >= Frequency(i))
            BPF_range = [BPF_range,Velocity(i)];
        end
    end
    BPF = max(BPF_range);
    %Find the fundamental frequency given some tolerances in frequency
 values
    %(1.5%)
    fundFreq_val = shaft_rotation/60;
    fundFreq_upper = fundFreq_val+(fundFreq_val*0.015);
    fundFreq_lower = fundFreq_val-(fundFreq_val*0.015);
    fundFreq_range = [];
    for i = 1:length(Frequency)
        if (Frequency(i) >= fundFreq_lower) && (fundFreq_upper >=
 Frequency(i))
            fundFreq_range = [fundFreq_range,Velocity(i)];
        end
    end
    fundFreq = max(fundFreq_range);
    %Find the second harmonic given some tolerances in frequency
 values
    %(1.5%)
    sec_harm_val = 2*fundFreq_val;
```

```
    sec_harm_upper = sec_harm_val+(sec_harm_val*0.015);
    sec_harm_lower = sec_harm_val-(sec_harm_val*0.015);
    sec_harm_range = [];
    for i = 1:length(Frequency)
        if (Frequency(i) >= sec_harm_lower) && (sec_harm_upper >=
Frequency(i))
            sec_harm_range = [sec_harm_range,Velocity(i)];
        end
    end
    sec_harm = max(sec_harm_range);
    %Find the third harmonic given some tolerances in frequency values
    %(1.5%)
    third_harm_val = 3*fundFreq_val;
    third_harm_upper = third_harm_val+(third_harm_val*0.015);
    third_harm_lower = third_harm_val-(third_harm_val*0.015);
    third_harm_range = [];
    for i = 1:length(Frequency)
        if (Frequency(i) >= third_harm_lower) && (third_harm_upper >=
Frequency(i))
            third_harm_range = [third_harm_range,Velocity(i)];
        end
    end
    third_harm = max(third_harm_range);
    %Find the fourth harmonic given some tolerances in frequency
values
    %(1.5%)
    fourth_harm_val = 4*fundFreq_val;
    fourth_harm_upper = fourth_harm_val+(fourth_harm_val*0.015);
    fourth_harm_lower = fourth_harm_val-(fourth_harm_val*0.015);
    fourth_harm_range = [];
    for i = 1:length(Frequency)
        if (Frequency(i) >= fourth_harm_lower) && (fourth_harm_upper
>= Frequency(i))
            fourth_harm_range = [fourth_harm_range,Velocity(i)];
        end
    end
    fourth_harm = max(fourth_harm_range);
    %Find the fifth harmonic given some tolerances in frequency values
    %(1.5%)
    fifth_harm_val = 5*fundFreq_val;
    fifth_harm_upper = fifth_harm_val+(fifth_harm_val*0.015);
    fifth_harm_lower = fifth_harm_val-(fifth_harm_val*0.015);
    fifth_harm_range = [];
    for i = 1:length(Frequency)
        if (Frequency(i) >= fifth_harm_lower) && (fifth_harm_upper >=
Frequency(i))
            fifth_harm_range = [fifth_harm_range,Velocity(i)];
        end
    end
    fifth_harm = max(fifth_harm_range);
    %Find the sixth harmonic given some tolerances in frequency values
    %(1.5%)
    sixth_harm_val = 6*fundFreq_val;
    sixth_harm_upper = sixth_harm_val+(sixth_harm_val*0.015);
```

```
    sixth_harm_lower = sixth_harm_val-(sixth_harm_val*0.015);
    sixth_harm_range = [];
    for i = 1:length(Frequency)
        if (Frequency(i) >= sixth_harm_lower) && (sixth_harm_upper >=
Frequency(i))
            sixth_harm_range = [sixth_harm_range,Velocity(i)];
        end
    end
    sixth_harm = max(sixth_harm_range);
    %Find the seventh harmonic given some tolerances in frequency
values
    %(1.5%)
    seventh_harm_val = 7*fundFreq_val;
    seventh_harm_upper = seventh_harm_val+(seventh_harm_val*0.015);
    seventh_harm_lower = seventh_harm_val-(seventh_harm_val*0.015);
    seventh_harm_range = [];
    for i = 1:length(Frequency)
        if (Frequency(i) >= seventh_harm_lower) && (seventh_harm_upper
>= Frequency(i))
            seventh_harm_range = [seventh_harm_range,Velocity(i)];
        end
    end
    seventh_harm = max(seventh_harm_range);
    %Find the eigth harmonic given some tolerances in frequency values
    %(1.5%)
    eighth_harm_val = 8*fundFreq_val;
    eighth_harm_upper = eighth_harm_val+(eighth_harm_val*0.015);
    eighth_harm_lower = eighth_harm_val-(eighth_harm_val*0.015);
    eighth_harm_range = [];
    for i = 1:length(Frequency)
        if (Frequency(i) >= eighth_harm_lower) && (eighth_harm_upper
>= Frequency(i))
            eighth_harm_range = [eighth_harm_range,Velocity(i)];
        end
    end
    eighth_harm = max(eighth_harm_range);
    %Find the twelfth harmonic given some tolerances in frequency
values
    %(1.5%)
    twelfth_harm_val = 12*fundFreq_val;
    twelfth_harm_upper = twelfth_harm_val+(twelfth_harm_val*0.015);
    twelfth_harm_lower = twelfth_harm_val-(twelfth_harm_val*0.015);
    twelfth_harm_range = [];
    for i = 1:length(Frequency)
        if (Frequency(i) >= twelfth_harm_lower) && (twelfth_harm_upper
>= Frequency(i))
            twelfth_harm_range = [twelfth_harm_range,Velocity(i)];
        end
    end
    twelfth_harm = max(twelfth_harm_range);
    %Find the twentieth harmonic given some tolerances in frequency
values
    %(1.5%)
    twentieth_harm_val = 20*fundFreq_val;
```

```
    twentieth_harm_upper = twentieth_harm_val
+(twentieth_harm_val*0.015);
    twentieth_harm_lower = twentieth_harm_val-
(twentieth_harm_val*0.015);
    twentieth_harm_range = [];
    for i = 1:length(Frequency)
        if (Frequency(i) >= twentieth_harm_lower) &&
 (twentieth_harm_upper >= Frequency(i))
            twentieth_harm_range = [twentieth_harm_range,Velocity(i)];
        end
    end
    twentieth_harm = max(twentieth_harm_range);
    %Find the thirty-sixth harmonic given some tolerances in frequency
 values
    %(1%)
    thirtysixth_harm_val = 36.6*fundFreq_val; %Recall a multiple of
 36.6
    thirtysixth_harm_upper = thirtysixth_harm_val
+(thirtysixth_harm_val*0.015);
    thirtysixth_harm_lower = thirtysixth_harm_val-
(thirtysixth_harm_val*0.015);
    thirtysixth_harm_range = [];
    for i = 1:length(Frequency)
        if (Frequency(i) >= thirtysixth_harm_lower) &&
 (thirtysixth_harm_upper >= Frequency(i))
            thirtysixth_harm_range =
 [thirtysixth_harm_range,Velocity(i)];
        end
    end
    thirtysixth_harm = max(thirtysixth_harm_range);
    %Find the thirty-seventh harmonic given some tolerances in
 frequency values
    %(1%)
    thirtyseventh_harm_val = 37.6*fundFreq_val; %Recall a multiple of
 37.6
    thirtyseventh_harm_upper = thirtyseventh_harm_val
+(thirtyseventh_harm_val*0.015);
    thirtyseventh_harm_lower = thirtyseventh_harm_val-
(thirtyseventh_harm_val*0.015);
    thirtyseventh_harm_range = [];
    for i = 1:length(Frequency)
        if (Frequency(i) >= thirtyseventh_harm_lower) &&...
                (thirtyseventh_harm_upper >= Frequency(i))
            thirtyseventh_harm_range =
 [thirtyseventh_harm_range,Velocity(i)];
        end
    end
    thirtyseventh_harm = max(thirtyseventh_harm_range);
    % Create vector of samples
    Samples = 0:length(Velocity)-1;
    Samples = Samples';
    %Combine all data
    allData =
 array2table(horzcat(Samples,Frequency,Velocity),'VariableNames',...
```

```
        {'Sample','Frequency_Hz','Velocity_mms'});
    % 3. SAVE DATA
    freqData =
 [BPF,fundFreq,sec_harm,third_harm,fourth_harm,fifth_harm,...

 sixth_harm,seventh_harm,eighth_harm,twelfth_harm,twentieth_harm,...
        thirtysixth_harm,thirtyseventh_harm]';
    dataNames =
 {'BPF','fundFreq','sec_harm','third_harm','fourth_harm',...
        'fifth_harm','sixth_harm','seventh_harm','eighth_harm',...

  'twelfth_harm','twentieth_harm','thirtysixth_harm','thirtyseventh_harm'}';
    writetable(allData,strcat(outputFilename,'.xls'));
    xlswrite(outputFilename,dataNames,'Sheet2','A1:A13');
    xlswrite(outputFilename,freqData,'Sheet2','B1:B13');
end
```

Published with MATLAB® R2018b

```
function []=processWfm()
% processWfm converts a velocity waveform dataset from an excel
 spreadsheet into ML features
% Refer to relevant Thesis/publications
% Outputs are saved in a spreadsheet file

    % CALCULATIONS:
    % 1. IMPORT DATA
    filename = 'your_filename';
    outputSuffix = '_StatParams';
    outputFilename = strcat(filename,outputSuffix);
    data = xlsread(filename,'A15:B4110');
    Time = data(:,1);
    Velocity = data(:,2);
    % 2. PROCESS DATA
    Accel = gradient(Velocity,0.000488); %0.000488 is the sampling
 rate
    % Convert to g
    Accel = Accel*0.000102;
    % Create vector of samples
    Samples = 0:length(Accel)-1;
    Samples = Samples';
    %Combine all data
    allData = array2table(horzcat(Velocity,Accel,Time,Samples),...
        'VariableNames',
{'Velocity_mms','Acceleration_g','Time_s','Sample'});
    %Plot acceleration data
    plot(Samples,Accel);
    %Calculate statistical parameters of acceleration data
    Mean = mean(Accel);
    StdDev = std(Accel);
    StdError = StdDev/sqrt(length(Accel));
    Median = median(Accel);
    Variance = var(Accel);
    Skew = skewness(Accel);
    Kurt = kurtosis(Accel);
    Range = range(Accel);
    MinVal = min(Accel);
    MaxVal = max(Accel);
    SumVals = sum(Accel);
    % 3. SAVE DATA
    Stats = [Mean,StdDev,StdError,Median,Variance,Skew,Kurt,Range,...
        MinVal,MaxVal,SumVals]';
    StatsNames = {'Mean','StdDev','StdError','Median','Variance',...
        'Skewness','Kurtosis','Range','MinVal','MaxVal','SumVals'}';
    writetable(allData,strcat(outputFilename,'.xls'));
    xlswrite(outputFilename,StatsNames,'Sheet2','A1:A11');
    xlswrite(outputFilename,Stats,'Sheet2','B1:B11');
end
```

Published with MATLAB® R2018b

C.2 Maintenance System Software

The present authors acknowledge the non-original functions used in the maintenance system software in the first section, followed by the presentation of all original and non-original functions in the second section. These were used to develop the pump RBM system in Chapters 3 and 4.

C.2.1 Function Authorship

All functions which have not been developed in the process of the present work are listed in alphabetical order and their original works and authors cited in Table C-1. The present authors sincerely appreciate the contribution of the original authors toward developing the present maintenance system software which includes their functions.

Table C-1: Non-Original Function Citations

FUNCTION NAME	CITATION(S) OF ORIGINAL WORK
estim_k.m	Nikolova, N. D., & Tenekedjiev, K. (2013). Numerical Version of the Non-Uniform Method for Finding Point Estimates of Uncertain Scaling Constants. In B. Igelnik, & Zurada, J.M. (Ed.), *Efficiency and Scalability Methods for Computational Intellect* (pp. 259 - 292). Hershey, Pennsylvania, United States of America: Information Science Reference. Nikolova, N. D., Mednikarov, B., Tenekedjiev, K. . (2012, 3 - 7 October). *Analytical Version of the Non-Uniform Method for Analysing Uncertain Scaling Constants.* Paper presented at the International Conference Automatics and Informatics, Sofia, Bulgaria.
optparam_udec.m	Nikolova, N., Mednikarov, B., & Tenekedjiev, K. (2018). Local Risk Proneness in Analytically Approximated Utility Functions Under Monotonically Decreasing Preferences. *Comptes Rendus des Academie Bulgare Des Sciences, 78*(11), 1534-1543.
optparam_xdec.m	
universal_utility_dec.m	
universal_utility_inv_dec.m	

C.2.2 The Epsilon Correction Function

As part of the work conducted in the present Thesis, the original function epsilon_correction.m was developed. The function is presented as part of the software behind the maintenance system in the following Section C.2.3. In the present section, the principles behind its development are discussed alongside its utilization in future systems.

C.2.2.1 Purpose of Epsilon Correction

Using the present maintenance system as an example; when the probabilities output from a Bayesian supervised classification algorithm are passed to decision trees, one of three possible scenarios can arise:

1. All probabilities are non-zero values
2. One or more of the probability values is zero
3. Neither of scenarios 1 or 2 occur

The analyst may not be aware of which situation is the true case. If scenarios 1 or 3 occur, there are encounter no issues as the decision trees can be calculated. However, when scenario 2 arises, calculation cannot proceed.

Zero-probabilities may arise due to errors in the learning data, errors in algorithm parameters or improper algorithm selection. The confusion, certainty and doubt matrices were discussed previously in Chapter 2. It was suggested that these tools may be used to help estimate the performance of classifiers arising due to modifications to learning data or algorithm parameters, or to compare multiple algorithms.

However, zero-probabilities may also legitimately arise. At this point, one of several approaches may be used to enable calculations to continue. For example, it is possible to assign the input vector to some unknown class, although this may add complexity to the classifier. Visualization techniques may be used to identify the most probable class or classes for the input vector provided that the data can be accurately represented in 3-dimensional space. It is also

possible to modify some combination of the classifier and the decision trees, although this would involve a time-consuming trial-and-error approach. The epsilon correction function was developed within the present work as a simple alternative to adjust zero-probabilities to enable further calculations.

C.2.2.2 Epsilon Correction Concept

The epsilon correction function adjusts zero-probabilities using the value specified as ε which is a small but non-zero value. This is analogous to the transformation of a vector into the objective space which was performed during the development of the linear discriminant classifiers in Chapter 4. Using this example, ε is the length of the zero-values in the objective space and accordingly non-zero values must be adjusted, as the total length of the vector must remain equal to one if the values are probabilities. A transformation of a vector of probabilities into another coordinate system should not alter its nature. In designing the RBM system, it was also decided that all class information is important and should be included in all decision tree calculations. Inclusion is ensured by setting the zero-probability values to ε.

The epsilon correction function takes as an input a vector of any number of probabilities, and transforms it into an ε-normalised version where n zero-probability values have been replaced by some small value ε and the non-zero-probability values have been adjusted by $(1-n\varepsilon)$.

C.2.2.3 Implications

It is suggested that the epsilon correction function is used in future data-driven maintenance systems which generate probabilities as a simple method to rectify the issue of zero-probability values. This ensures that future systems include all possible information in analyses which follow the calculation of probabilities.

C.2.3 Software

The present section contains all original and non-original software developed for the shipboard pump maintenance system application as described in Chapters 3 and 4. Refer to Section C.2.1 for the list of non-original functions and their appropriate citations. 48 functions are listed below and presented in alphabetical order. The software is initialized using main_menu.m.

arctan_function.m
bayesian_classifier_learning.m
bayesian_classifier_measurement.m
bayesian_recognition_single.m
calculation_menu.m
chcknint.m
chcknmat.m
compound_lottery_A1.m
decision_tree_pump.m
decision_tree_test.m
display_values.m
epsilon_correction.m
estim_K.m
eval_tree_atsea_emergency_pump.m
eval_tree_atsea_emergency_test.m
eval_tree_atsea_pump.m
eval_tree_atsea_test.m
eval_tree_atwharf_enginesrunning_pump.m
eval_tree_atwharf_enginesrunning_test.m
eval_tree_atwharf_norunning_pump.m
eval_tree_atwharf_norunning_test.m
evaluate_MAU_pump.m
evaluate_MAU_test.m
generate_noise.m
learning_database.m
main_menu.m
mean_distance.m
measurement_database.m
multi_attribute_utilities_atsea_pump.m
multi_attribute_utilities_atsea_test.m
multi_attribute_utilities_atwharf_pump.m
multi_attribute_utilities_atwharf_test.m
objective_space.m
optparam_udec.m
optparam_xdec.m
Param_estim.m
performance_est.m
probability_matrix.m
recursive_ICOL.m
replace_values.m
select_policy_noisy.m
select_policy_pump.m
select_policy_test.m
simple_lottery.m
subj_prob_database.m
universal_utility_dec.m
universal_utility_inv_dec.m
util_rng_database.m

```
function [y] = arctan_function(a,b,x,xu,xd)
%arctan_function evaluates the decreasing arctan utility function as y
 for the given inputs
%
%   Inputs: a   Fitted parameter double
%           b   Fitted parameter double
%           xu  Upper bound on x double
%           xd  Lower bound on x double
%
%   Output: y   The arctan function position of x double
%

% CALCULATIONS:
    top = atan((a*x) - (a*b)) - atan((a*xd)-(a*b));
    bottom = atan((a*xu) - (a*b))- atan((a*xd)-(a*b));
    y = 1 - (top/bottom);
end
```

Published with MATLAB® R2018b

```
function [wlnw_c,w0lnw_c,wlPw_c,w0lPw_c]=
 bayesian_classifier_learning(num_data)
%bayesian_classifier_learning learns linear classifier parameters from
 input data
%
%   Inputs:  num_data           Classifier selection parameter as
 1,2,3 or 4 double
%
%   Outputs: wlnw_c, w01nw_c    Cell matrices of linear classifier
%                               parameters, reference Duda and
 Hart(2012),
%                               non-prior weighted
%            wlPw_c, w01Pw_c    Cell matrices of linear classifier
%                               parameters, reference Duda and
 Hart(2012),
%                               prior weighted
%
% Outputs are also saved to a file specified as \classifiers

% CALCULATIONS:
    inp_filename1_str=['DATA' int2str(num_data) '\parameters'];
    load(inp_filename1_str,'mu_c');
    inp_filename2_str=['DATA' int2str(num_data) '\objs'];
    load(inp_filename2_str,'T_m','off_v','Priors_v','Snw_m','SPw_m');
    c=length(mu_c);
    [wlnw_c,w0lnw_c,wlPw_c,w0lPw_c]=deal(cell(1,c));
    Snw_obj_m=T_m*Snw_m*T_m';
    SPw_obj_m=T_m*SPw_m*T_m';
    for k=1:c
        muk_v=mu_c{k};
        muk_obj_v=T_m*muk_v-off_v;
        wlnw_c{k}=Snw_obj_m\muk_obj_v;
        w0lnw_c{k}=log(Priors_v(k))-muk_obj_v'*wlnw_c{k}/2;
        wlPw_c{k}=SPw_obj_m\muk_obj_v;
        w0lPw_c{k}=log(Priors_v(k))-muk_obj_v'*wlPw_c{k}/2;
    end
    out_filename1_str=['DATA' int2str(num_data) '\classifiers'];
    save(out_filename1_str,'wlnw_c','w0lnw_c','wlPw_c','w0lPw_c');
end
```

Published with MATLAB® R2018b

```
function [Meas_Res_Sampleknw_v,Meas_Res_SamplekPw_v]=
 bayesian_classifier_measurement(flag_dat,x_v)
% bayesian_classifier_measurement classifies x_v input vector using
 prebuilt classifiers
%
%   Inputs:  flag_dat              Classifier selection parameter as
 1,2,3 or 4 double
%            x_v                   Vector input for classification
%
%   Outputs: Meas_Res_Sampleknw_v  Vector of class posterior
 probabilities
%                                  determined from x_v,non-prior
 weighted
%            Meas_Res_SamplekPw_v  Vector of class posterior
 probabilities,
%                                  determined from x_v,prior weighted

% CALCULATIONS:
    inp_filename2_str=['DATA' int2str(flag_dat) '\objs'];
    load(inp_filename2_str,'T_m','off_v');
    inp_filename3_str=['DATA' int2str(flag_dat) '\classifiers'];
    load(inp_filename3_str,'wlnw_c','w0lnw_c','wlPw_c','w0lPw_c');
    c=length(wlnw_c);
    d_obj=size(wlnw_c{1},1);
    wlnw_m=NaN(d_obj,c);
    w0lnw_v=NaN(1,c);
    wlPw_m=NaN(d_obj,c);
    w0lPw_v=NaN(1,c);
    for k=1:c
        wlnw_m(:,k)=wlnw_c{k};
        w0lnw_v(k)=w0lnw_c{k};
        wlPw_m(:,k)=wlPw_c{k};
        w0lPw_v(k)=w0lPw_c{k};
    end
    [Posnw_v,PosPw_v]=
 bayesian_recognition_single(x_v,T_m,off_v,wlnw_m,w0lnw_v,wlPw_m,w0lPw_v);
    Meas_Res_Sampleknw_v =Posnw_v';
    Meas_Res_SamplekPw_v=PosPw_v';
end
```

Published with MATLAB® R2018b

```
function [Posnw_v,PosPw_v]= bayesian_recognition_single(...
    x_v,T_m,off_v,wlnw_m,w0lnw_v,wlPw_m,w0lPw_v)
% bayesian_recognition_single- recognises a single input vector using
 one discriminant function
%
%   Inputs:  x_v                    Vector input for classification
%            T_m,off_v,wlnw_m,      Classifier parameters, refer to
%            w01nw_v,wlPw_m         Duda and Hart (2012)
%            w01Pw_v
%
%   Outputs: Posnw_v                Class posterior probability double
%                                   determined from x_v,non-prior
 weighted
%            PosPw_v                Class posterior probability double
%                                   determined from x_v,prior weighted

% CALCULATIONS:
    x_obj_v=T_m*x_v-off_v;
    c=length(w0lnw_v);
    [gnw_v,gPw_v,Posnw_v,PosPw_v]=deal(NaN(c,1));
    for k=1:c
        gnw_v(k)=wlnw_m(:,k)'*x_obj_v+w0lnw_v(k);
        gPw_v(k)=wlPw_m(:,k)'*x_obj_v+w0lPw_v(k);
    end
    for k=1:c
        Posnw_v(k)=1/sum(exp(gnw_v-gnw_v(k)));
        PosPw_v(k)=1/sum(exp(gPw_v-gPw_v(k)));
    end
end
```

Published with MATLAB® R2018b

```
function [] = calculation_menu()
% calculation_menu loads data and prompts user for inputs to calculate
 a policy
%
%   Function is called from main_menu() and passes to
 select_policy(...)
%
%   The results of calculations performed within this function are
 saved in
%   output files specified by the user
%

% CALCULATIONS:
    % Define the choice variable
    choice = 0;
    while choice ~= 4
        choice = menu('Calculation Menu. Please select from the
 following:',...
            '1 = Load Pump Measurement data to calculate policy',...
            '2 = Load Test Measurement data to calculate policy',...
            '3 = Generate Noisy Measurement Data based on Learning
 Samples and calculate policy',...
            '4 = Return to Main Menu (eg. For changing/updating system
 parameters!)');
       if choice == 1
            %Load data
            inp_filename_str = ['meas' '\' 'pumpdata'];
            load(inp_filename_str,'pumpdata_m');
            %Display properties (size) of data
            num_rows = size(pumpdata_m,1);
            num_columns = size(pumpdata_m,2);
            disp(['Loaded Pump Measurement Data. Dataset is ...',...
                num2str(num_rows),'... rows by ...',...
                num2str(num_columns),'...columns.']);
            disp('Rows represent each measurement in this case.');
            %Display a sample of the data
            disp('Displaying two rows as an example of two
 measurements:');
            example_m = pumpdata_m(1:2,:);
            disp(example_m);
            %Confirm wanting to calculate
            calculate = 'n'; %#ok<NASGU>
            calculate = input('Proceed with calculation using
 specified pump measurement dataset? Y/N as character');
            if (calculate == 'Y') || (calculate == 'y')
                test_name_str = input('Please enter a unique test name
 for results file as a string:');
                %Prompt user to select situation for calculations
                selection = menu('Please select:','Vessel is at the
 wharf, no equipment running'...
                    ,'Vessel is at the wharf, other machinery is
 running'...
```

```
                    ,'Vessel is slow steaming in the harbour and it is
not an emergency'...
                    ,'Vessel is slow steaming in the harbour and it is
an emergency');
                % Change selection into flag_dat and emergency
variables
                flag_dat = selection;
                % Load remaining variables and calculate
                %Load trained prior probabilities
                inp_filename1_str = ['DATA'
num2str(flag_dat) '\trained_Ppr'];
                load(inp_filename1_str,'trained_Ppr_v');
                disp('Loaded trained prior probabilities:');
                disp(trained_Ppr_v);
                %Create matrices to store results
                classified_measurements_m = pumpdata_m;
                Meas_Res_Sampleknw_m = zeros(num_rows,8);
                policynw_v = zeros(num_rows,1);
                policynw_str_c = cell(num_rows,1);
                Meas_Res_SamplekPw_m = zeros(num_rows,8);
                policyPw_v = zeros(num_rows,1);
                policyPw_str_c = cell(num_rows,1);
                results_labels_c = {'classified_measurements_m',...
                    'Meas_Res_Sampleknw_m','policynw_v',...
                    'policynw_str_c','Meas_Res_SamplekPw_m',...
                    'policyPw_v','policyPw_str_c'};
                %Calculate
                disp('Calculating ...');
                for i = 1:num_rows
                    x_v = pumpdata_m(i,:)';
                    [Meas_Res_Sampleknw_m(i,:),policynw_v(i),...

policynw_str_c{i},Meas_Res_SamplekPw_m(i,:),...
                        policyPw_v(i),policyPw_str_c{i}] =...

select_policy_pump(x_v,trained_Ppr_v,selection);
                end
                %Save results
                numeric_results_m = [classified_measurements_m,...
                    Meas_Res_Sampleknw_m,policynw_v,...
                    Meas_Res_SamplekPw_m,policyPw_v];
                all_results_t = table(classified_measurements_m,...
                    Meas_Res_Sampleknw_m,policynw_v,policynw_str_c,...
                    Meas_Res_SamplekPw_m,policyPw_v,policyPw_str_c,...
                    'VariableNames',results_labels_c);
                out_filename1_str = ['results\','SLCT_',...
                    num2str(flag_dat),'_pump_',test_name_str,...
                    '_results_',num2str(floor(now))];
                save(out_filename1_str,'numeric_results_m',...
                    'all_results_t');
                disp(head(all_results_t));
                disp(['Saved results in: ',out_filename1_str]);
            elseif (calculate == 'N') || (calculate == 'n')
                disp('Returning to Calculation Menu.')
```

```
            end
        elseif choice == 2 %Select Test Data
            %Load data
            inp_filename_str = ['meas' '\' 'testdata'];
            load(inp_filename_str,'testdata_m');
            testdata_m = testdata_m';
            %Display properties (size) of data
            num_rows = size(testdata_m,1);
            num_columns = size(testdata_m,2);
            disp(['Loaded Test Measurement Data. Dataset is ...',...
                num2str(num_rows),'... rows by ...',...
                num2str(num_columns),'...columns.']);
            disp('Rows represent each measurement in this case.');
            %Display a sample of the data
            disp('Displaying two rows as an example of two
measurements:');
            example_m = testdata_m(1:2,:);
            disp(example_m);
            %Confirm wanting to calculate
            calculate = 'n'; %#ok<NASGU>
            calculate = input('Proceed with calculation using
specified test measurement dataset? Y/N as character');
            if (calculate == 'Y') || (calculate == 'y')
                test_name_str = input('Please enter a unique test name
for results file as a string:');
                %Prompt user to select situation for calculations
                selection = menu('Please select:','Vessel is at the
wharf, no equipment running'...
                    ,'Vessel is at the wharf, other machinery is
running'...
                    ,'Vessel is slow steaming in the harbour and it is
not an emergency'...
                    ,'Vessel is slow steaming in the harbour and it is
an emergency');
                % Change selection into flag_dat and emergency
variables
                flag_dat = selection;
                % Load remaining variables and calculate
                %Load trained prior probabilities
                inp_filename1_str = ['DATA'
num2str(flag_dat) '\trained_Ppr'];
                load(inp_filename1_str,'trained_Ppr_v');
                disp('Loaded trained prior probabilities:');
                disp(trained_Ppr_v);
                %Create matrices to store results
                classified_measurements_m = testdata_m;
                Meas_Res_Sampleknw_m = zeros(num_rows,8);
                policynw_v = zeros(num_rows,1);
                policynw_str_c = cell(num_rows,1);
                Meas_Res_SamplekPw_m = zeros(num_rows,8);
                policyPw_v = zeros(num_rows,1);
                policyPw_str_c = cell(num_rows,1);
                results_labels_c = {'classified_measurements_m',...
                    'Meas_Res_Sampleknw_m','policynw_v',...
```

```
                    'policynw_str_c','Meas_Res_SamplekPw_m',...
                    'policyPw_v','policyPw_str_c'};
                %Calculate
                disp('Calculating ...');
                for i = 1:num_rows
                    x_v = testdata_m(i,:)';
                    [Meas_Res_Sampleknw_m(i,:),policynw_v(i),...

policynw_str_c{i},Meas_Res_SamplekPw_m(i,:),...
                        policyPw_v(i),policyPw_str_c{i}] =...

select_policy_test(x_v,trained_Ppr_v,selection);
                end
                %Save results
                numeric_results_m = [classified_measurements_m,...
                    Meas_Res_Sampleknw_m,policynw_v,...
                    Meas_Res_SamplekPw_m,policyPw_v];
                all_results_t = table(classified_measurements_m,...
                    Meas_Res_Sampleknw_m,policynw_v,policynw_str_c,...
                    Meas_Res_SamplekPw_m,policyPw_v,policyPw_str_c,...
                    'VariableNames',results_labels_c);
                out_filename1_str = ['results\','SLCT_',...
                    num2str(flag_dat),'_test_',test_name_str,...
                    '_results_',num2str(floor(now))];
                save(out_filename1_str,'numeric_results_m',...
                    'all_results_t');
                disp(head(all_results_t));
                disp(['Saved results in: ',out_filename1_str]);
            elseif (calculate == 'N') || (calculate == 'n')
                disp('Returning to Calculation Menu.')
            end
        elseif choice == 3 %Generate some noisy Data
            %Prompt user to select situation for calculations
            selection = menu('Please select:','Vessel is at the wharf,
no equipment running'...
                ,'Vessel is at the wharf, other machinery is
running'...
                ,'Vessel is slow steaming in the harbour and it is not
an emergency'...
                ,'Vessel is slow steaming in the harbour and it is an
emergency');
            % Change selection into flag_dat and emergency variables
            flag_dat = selection;
            %Generate data
            noise = input('Please enter the percentage of noise to be
added to the learning data as an number greater than zero:');
            %Produce the noisy data for classfication
            [x_noisy_c] = generate_noise(flag_dat,noise);
            %Dataset for all noisy data to be classified, pull from
cell
            %array
            noisydata_m = [];
            for i = 1:length(x_noisy_c)
                noisydata_m = [noisydata_m,x_noisy_c{i}]; %#ok<AGROW>
```

```
            end
            noisydata_m = noisydata_m';
            %Display properties (size) of data
            num_rows = size(noisydata_m,1);
            num_columns = size(noisydata_m,2);
            disp(['Loaded Noisy Measurement Data. Dataset is ...',...
                num2str(num_rows),'... rows by ...',...
                num2str(num_columns),'...columns.']);
            disp('Rows represent each measurement in this case.');
            %Display a sample of the data
            disp('Displaying two rows as an example of two
measurements:');
            example_m = noisydata_m(1:2,:);
            disp(example_m);
            %Confirm wanting to calculate
            calculate = 'n'; %#ok<NASGU>
            calculate = input('Proceed with calculation using
specified noisy measurement dataset? Y/N as character');
            if (calculate == 'Y') || (calculate == 'y')
                test_name_str = input('Please enter a unique test name
for results file as a string:');
                % Load remaining variables and calculate
                %Load trained prior probabilities
                inp_filename1_str = ['DATA'
num2str(flag_dat) '\trained_Ppr'];
                load(inp_filename1_str,'trained_Ppr_v');
                disp('Loaded trained prior probabilities:');
                disp(trained_Ppr_v);
                %Create matrices to store results
                classified_measurements_m = noisydata_m;
                Meas_Res_Sampleknw_m = zeros(num_rows,8);
                policynw_v = zeros(num_rows,1);
                policynw_str_c = cell(num_rows,1);
                Meas_Res_SamplekPw_m = zeros(num_rows,8);
                policyPw_v = zeros(num_rows,1);
                policyPw_str_c = cell(num_rows,1);
                results_labels_c = {'classified_measurements_m',...
                    'Meas_Res_Sampleknw_m','policynw_v',...
                    'policynw_str_c','Meas_Res_SamplekPw_m',...
                    'policyPw_v','policyPw_str_c'};
                %Calculate
                disp('Calculating ...');
                for i = 1:num_rows
                    x_v = noisydata_m(i,:)';
                    [Meas_Res_Sampleknw_m(i,:),policynw_v(i),...

policynw_str_c{i},Meas_Res_SamplekPw_m(i,:),...
                        policyPw_v(i),policyPw_str_c{i}] = ...

select_policy_noisy(x_v,trained_Ppr_v,selection);
                end
                %Save results
                numeric_results_m = [classified_measurements_m,...
```

```
 Meas_Res_Sampleknw_m,policynw_v,Meas_Res_SamplekPw_m,policyPw_v];
                all_results_t = table(classified_measurements_m,...
                    Meas_Res_Sampleknw_m,policynw_v,policynw_str_c,...
                    Meas_Res_SamplekPw_m,policyPw_v,policyPw_str_c,...
                    'VariableNames',results_labels_c);
                out_filename1_str = ['results\','SLCT_',...
                    num2str(flag_dat),'_',num2str(noise),'_noise_',...
                    test_name_str,'_results_',num2str(floor(now))];

 save(out_filename1_str,'numeric_results_m','all_results_t');
                disp(head(all_results_t));
                disp(['Saved results in: ',out_filename1_str]);
            elseif (calculate == 'N') || (calculate == 'n')
                disp('Returning to Calculation Menu.')
            end
        end
        disp('Returning to Main Menu.');
    end
end
```

Published with MATLAB® R2018b

```
function [errorcode,n,str]=chcknint(s,a,defaultn)
% CHCKNINT Checks whether the input argument S is a proper
 representation of a real integer number.
%
% [errorcode,n,str]=chcknint(s,a,defaultn)
%
% S         is a variable to be checked whether it is a real integer
 number.
% A         is either a multi-dimensional integer vector, representing
 the domain D of N,
%           or an integer value, representing the lower margin of the
 domain D of N
%           or an empty matrix (by default) meaning that the domain D
 is all integer numbers
% DEFAULTN  is an integer number representing the default for N if S
 is empty.
%           To use default value of A you may pass in the empty matrix
 ([]).
%
% ERRORCODE is flag for abnormal termination of 'CHCKNINT'.
%           ERRORCODE is 0 when N is eligible output.
%           ERRORCODE is 1 when the first input parameter is not
 eligible.
%           ERRORCODE is 2 when the second input parameter is not
 eligible.
%           ERRORCODE is 3 when the third input parameter is not
 eligible.
%           ERRORCODE is 4 when the number of the input parameters is
 not eligible.
% N         is an integer value from the domain A or is empty if
 ERRORCODE is not 0
% STR       is a text string containing message from 'CHCKNINT' if
 ERRORCODE is not 0
%

% CALCULATIONS:
if nargin==1
    a=[];
    defaultn=[];
elseif nargin==2
    defaultn=[];
elseif nargin<1
    errorcode=4;
    str='incorrect number of input parameters (message from
 CHCKNINT)';
    n=[];
    return
end
er=0;
if isempty(a)
    a=[];
elseif ndims(a)~=2
```

```
    er=1;
elseif all(size(a)~=1)
 er=2;
elseif isnumeric(a)~=1
 er=3;
elseif any(isfinite(a)~=1)
 er=4;
elseif any(isreal(a)~=1)
    er=5;
elseif any(a~=ceil(a))
    er=6;
end
if er~=0
    errorcode=2;
    str='incorrect input parameter A (message from CHCKNINT)';
    n=[];
    return
end
er=0;
if isempty(defaultn)
    defaultn=[];
elseif ndims(defaultn)~=2
    er=1;
elseif length(defaultn)~=1
    er=2;
elseif isnumeric(defaultn)~=1
    er=3;
elseif isfinite(defaultn)~=1
    er=4;
elseif isreal(defaultn)~=1
    er=5;
elseif defaultn~=ceil(defaultn)
    er=6;
elseif ~isempty(a)
    if length(a)==1
        if defaultn<a
            er=7;
        end
    elseif length(a)>1
        if all(a~=defaultn)
            er=8;
        end
    end
end
if er~=0
    errorcode=3;
    str='incorrect input parameter DEFAULTN (message from CHCKNINT)';
    n=[];
    return
end
if isempty(s) & ~isempty(defaultn)
    n=defaultn;
    errorcode=0;
    str=[];
```

```
    return
end
n=s;
er=0;
if isempty(n)
    er=1;
elseif ndims(n)~=2
    er=2;
elseif length(n)~=1
    er=3;
elseif isnumeric(n)~=1
    er=4;
elseif isfinite(n)~=1
    er=5;
elseif isreal(n)~=1
    er=6;
elseif n~=ceil(n)
    er=7;
elseif ~isempty(a)
    if length(a)==1
        if n<a
            er=8;
        end
    elseif length(a)>1
        if all(a~=n)
            er=9;
        end
    end
end
if er~=0
    errorcode=1;
    str='incorrect input parameter S (message from CHCKNINT)';
    n=[];
    return
end
errorcode=0;
str=[];
end
```

Published with MATLAB® R2018b

```
function [errorcode,nmat,str]=chcknmat(s,m,n,defaultx)
% CHCKNMAT Checks whether the input argument S is a [M x N] real
 numeric matrix.
%
% [errorcode,nmat,str]=chcknmat(s,m,n,defaultx)
%
% S         is a variable to be checked whether it is a real numeric
 matrix.
% M         is the number of the required rows of NMAT. By default M
 is 1
% N         is the number of the required columns of NMAT. By default
 N is 1
% DEFAULTX  is an [M x N] real numeric matrix representing the default
%           for NMAT if S is empty.
%           To use default values for S, M or N  you may pass in the
 empty matrix ([]).
%
% ERRORCODE is flag for abnormal termination of 'CHCKNMAT'.
%           ERRORCODE is 0 when NMAT is eligible output.
%           ERRORCODE is 1 when the first input parameter is not
 eligible.
%           ERRORCODE is 2 when the second input parameter is not
 eligible.
%           ERRORCODE is 3 when the third input parameter is not
 eligible.
%           ERRORCODE is 4 when the fourth input parameter is not
 eligible.
%           ERRORCODE is 5 when the number of the input parameters is
 not eligible.
% NMAT      is the [M x N] real numeric matrix or is empty if
 ERRORCODE is not 0
% STR       is a text string containing message from 'CHCKNMAT' if
 ERRORCODE is not 0
%

% CALCULATIONS:
if nargin==1
  m=1;
    n=1;
    defaultx=[];
elseif nargin==2
 n=1;
    defaultx=[];
elseif nargin==3
    defaultx=[];
elseif nargin<1
    errorcode=5;
    str='incorrect number of input parameters (message from
 CHCKNMAT)';
    nmat=[];
 return
end
```

```
[er,m]=chcknint(m,1,1);
if er~=0
    errorcode=2;
    str='incorrect input parameter M (message from CHCKNMAT)';
    nmat=[];
    return
end
[er,n]=chcknint(n,1,1);
if er~=0
    errorcode=3;
    str='incorrect input parameter N (message from CHCKNMAT)';
    nmat=[];
    return
end
er=0;
if isempty(defaultx)
    defaultx=[];
elseif ndims(defaultx)~=2
    er=1;
elseif any(size(defaultx)~=[m,n])
    er=2;
elseif isnumeric(defaultx)~=1
    er=3;
elseif any(any(isfinite(defaultx)~=1))
    er=4;
elseif any(any(isreal(defaultx)~=1))
    er=5;
end
if er~=0
    errorcode=4;
    str='incorrect input parameter DEFAULTX (message from CHCKNMAT)';
    nmat=[];
    return
end
if isempty(s) & ~isempty(defaultx)
    nmat=defaultx;
    errorcode=0;
    str=[];
    return
end
nmat=s;
er=0;
if isempty(nmat)
    er=1;
elseif ndims(nmat)~=2
    er=2;
elseif any(size(nmat)~=[m,n])
    er=3;
elseif isnumeric(nmat)~=1
    er=4;
elseif any(any(isfinite(nmat)~=1))
    er=5;
elseif any(any(isreal(nmat)~=1))
    er=6;
```

```
end
if er~=0
    errorcode=4;
    str='incorrect input parameter S (message from CHCKNMAT)';
    nmat=[];
    return
end
errorcode=0;
str=[];
end
```

Published with MATLAB® R2018b

```
function [exp_util]=compound_lottery_A1(Pos_v,...
    subjective_probability_bestcase_v,multi_attribute_utilities_v)
% compound_lottery_A1 calculates the expected utility of a compound
 lottery representing Action A1
%
% Inputs:  Pos_v                                8 x 1 vector of state
 probabilities
%          subjective_probability_bestcase_v   8 x 1 vector of
 subjective probabilities,
%                                              repair time per state
 less than 30 mins
%          multi_attribute_utilities_v         16 x 1 vector of
 utility values, two
%                                              elements per state of
 nature. Beginning
%                                              with state 1, we
 allocate positions
%                                              (1) and (2), then state
 2 in positions
%                                              (3) and (4).. etc.
%
% Output:  exp_util                             Expected utility double
%
% Further detail can be found in relevant Thesis and supporting
 publications

% CALCULATIONS:
    exp_util = 0;
    subjective_probability_worstcase_v = 1- ...
        subjective_probability_bestcase_v;
    for i = 1:length(Pos_v)
        exp_util = exp_util + ((Pos_v(i)*...
            subjective_probability_worstcase_v(i))*...
            multi_attribute_utilities_v((2*i)-1))...
            +((Pos_v(i)*subjective_probability_bestcase_v(i))*...
            multi_attribute_utilities_v(2*i));
    end
end
```

Published with MATLAB® R2018b

```
function [policynw,exp_utilnw_v,policyPw,exp_utilPw_v] =
 decision_tree_pump(flag_dat,Posnw_v,PosPw_v)
% decision_tree_pump calculates decision tree for Posnw_v and PosPw_v
% Calculations are based on linear discriminant classifier chosen
 using flag_dat
% The function loads utility values and attribute weights from
% survey_utils.mat which describe the pump for its calculations
%
% Inputs:  flag_dat      Integer equalling 1,2, 3 or 4 used to select
 which dataset
%                        was used to produce posterior probabilities
 Posnw_v and
%                        PosPw_v
%          emergency     Interger equalling 0 indicating no emergency
 or 1
%                        indicating an emergency (we cannot stop the
 pump)
%
%          Posnw_v       Posterior probabilities computed using non-
weighted
%                        covariance matrix using bayesian linear
 discriminant classifier
%
%          PosPw_v       Posterior probabilities computed using prior-
weighted
%                        covariance matrix using bayesian linear
 discriminant classifier
%
% Outputs: policynw      Policy or action number posessing the maximum
%                        expected utility using Posnw_v double
%
%         exp_utilnw_v   Expected utilities computed using decision
 tree and Posnw_v
%
%         policyPw       Policy or action number posessing the maximum
%                        expected utility using PosPw_v double
%
%         exp_utilPw_v    Expected utilities computed using decision
 tree and
%                         PosPw_v
%

%CALCULATIONS:
    if flag_dat == 1 % Calculate tree when vessel is at the wharf
                     % and NOTHING is running
        [exp_utilnw_v] =
 eval_tree_atwharf_norunning_pump(Posnw_v,flag_dat);
        [exp_utilPw_v] =
 eval_tree_atwharf_norunning_pump(PosPw_v,flag_dat);
    elseif flag_dat == 2 % Calculate tree when vessel is at the wharf
                         % and engines ARE running
```

```
        [exp_utilnw_v] =
 eval_tree_atwharf_enginesrunning_pump(Posnw_v,flag_dat);
        [exp_utilPw_v] =
 eval_tree_atwharf_enginesrunning_pump(PosPw_v,flag_dat);
    elseif (flag_dat == 3)% Calculate tree when vessel is moving in
 harbour
        [exp_utilnw_v] = eval_tree_atsea_pump(Posnw_v,flag_dat);
        [exp_utilPw_v] = eval_tree_atsea_pump(PosPw_v,flag_dat);
    elseif (flag_dat == 4)% Calculate tree when vessel is moving in
 harbour
        [exp_utilnw_v] =
 eval_tree_atsea_emergency_pump(Posnw_v,flag_dat);
        [exp_utilPw_v] =
 eval_tree_atsea_emergency_pump(PosPw_v,flag_dat);
    end
    [~,policynw]=max(exp_utilnw_v);
    [~,policyPw]=max(exp_utilPw_v);
end
```

Published with MATLAB® R2018b

```
function [policynw,exp_utilnw_v,policyPw,exp_utilPw_v] =
 decision_tree_test(flag_dat,Posnw_v,PosPw_v)
% decision_tree_test calculates decision tree for Posnw_v and PosPw_v
% Calculations are based on linear discriminant classifier chosen
 using flag_dat
% The function loads utility values and attribute weights from
% test_utils.mat for its calculations
%
% Inputs:  flag_dat      Integer equalling 1,2, 3 or 4 used to select
 which dataset
%                        was used to produce posterior probabilities
 Posnw_v and
%                        PosPw_v
%          emergency     Interger equalling 0 indicating no emergency
 or 1
%                        indicating an emergency (we cannot stop the
 pump)
%
%          Posnw_v       Posterior probabilities computed using non-
weighted
%                        covariance matrix using bayesian linear
 discriminant classifier
%
%          PosPw_v       Posterior probabilities computed using prior-
weighted
%                        covariance matrix using bayesian linear
 discriminant classifier
%
% Outputs: policynw      Policy or action number posessing the maximum
%                        expected utility using Posnw_v double
%
%         exp_utilnw_v   Expected utilities computed using decision
 tree and Posnw_v
%
%         policyPw       Policy or action number posessing the maximum
%                        expected utility using PosPw_v double
%
%         exp_utilPw_v    Expected utilities computed using decision
 tree and
%                         PosPw_v
%

%CALCULATIONS:
    if flag_dat == 1 % Calculate tree when vessel is at the wharf
                     % and NOTHING is running
        [exp_utilnw_v] =
 eval_tree_atwharf_norunning_test(Posnw_v,flag_dat);
        [exp_utilPw_v] =
 eval_tree_atwharf_norunning_test(PosPw_v,flag_dat);
    elseif flag_dat == 2 % Calculate tree when vessel is at the wharf
                         % and engines ARE running
```

```
        [exp_utilnw_v] =
 eval_tree_atwharf_enginesrunning_test(Posnw_v,flag_dat);
        [exp_utilPw_v] =
 eval_tree_atwharf_enginesrunning_test(PosPw_v,flag_dat);
    elseif (flag_dat == 3)% Calculate tree when vessel is moving in
 harbour
        [exp_utilnw_v] = eval_tree_atsea_test(Posnw_v,flag_dat);
        [exp_utilPw_v] = eval_tree_atsea_test(PosPw_v,flag_dat);
    elseif (flag_dat == 4)% Calculate tree when vessel is moving in
 harbour
        [exp_utilnw_v] =
 eval_tree_atsea_emergency_test(Posnw_v,flag_dat);
        [exp_utilPw_v] =
 eval_tree_atsea_emergency_test(PosPw_v,flag_dat);
    end
    [~,policynw]=max(exp_utilnw_v);
    [~,policyPw]=max(exp_utilPw_v);
end
```

Published with MATLAB® R2018b

```
function [dataset_m,new_dataset_m] =
 display_values(dataset_m,rows_v,columns_v)
% display_values searches through a dataset and finds values of
 interest
%   Inputs: dataset_m     The dataset matrix which has values
 requiring
%                         replacement
%              rows_v     The vector of row indices for values in
 dataset_m
%                         which are to be replaced, must have equal
 length as
%                         columns_v and new_values_v
%           columns_v     The vector of column indices for values in
 dataset_m
%                         which are to be replaced, must have equal
 length as
%                         rows_v and new_values_v
%
%  Outputs: dataset_m     The dataset matrix which has values
 requiring
%                         replacement
%       new_dataset_m     dataset_m with values replaced as specified
 with
%                         indices in rows_v and columns_v with values
 in
%                         dataset_m. All other values are zeros
%

%CALCULATIONS:
    new_dataset_m = zeros(size(dataset_m,1),size(dataset_m,2));
    for i = 1:length(rows_v)
        for j = 1:length(columns_v)

 new_dataset_m(rows_v(i),columns_v(j))=dataset_m(rows_v(i),columns_v(j));
        end
    end
end
```

Published with MATLAB® R2018b

```
function [corrected_probabilities_v,flagstop,str] =
 epsilon_correction(probabilities_v)
% epsilon_correction corrects vectors of probabilities where some are
 zero to make values small but non-zero
%   MATLAB displays some small values as zero when in fact they are
 not
%   zero. A small correction is necessary in the case of probabilities
 used
%   to perfrom linear discriminant analysis parameter calculations,
 which
%   lead to the development of this function. Output probabilities
 look
%   like zero but are non-zero so that calculations may proceed.
%   The function also implements error checking to ensure inputs are
%   probabilities.
%
% Inputs: probabilities_v            A 1 x n vector of probabilities
 where one or more elements are zero
%
% Outputs: corrected_probabilities_v  A 1 x n vector of probabilities
 corresponding to probabilities_v where one
%                                     or more elements are small non-
zero values

%CALCULATIONS:
    if nargin<1
        flagstop=2;
        str='Incorrect count of input parameters (message from
 ''epsilon_correction'')';
        corrected_probabilities_v=[];
        return
    end
    err=0;
    if ~ismatrix(probabilities_v)
        err=1;
    end
    if err==0
        m=size(probabilities_v,2);
        err=chcknmat(probabilities_v,1,m)*10;
    end
    if err==0 && any(probabilities_v<0)
        err=10;
    end
    if err==0 && sum(probabilities_v)~=1
        err=20;
    end
    if err>0
        flagstop=1;
        str='Incorrect input parameter probabilities_v (message from
 ''epsilon_correction'')';
        disp(['internal error code ' num2str(err)]);
        corrected_probabilities_v=[];
```

```
        return
    end
    % Define small value for modification
    epsi = 1*10^-20;
    fl0_v=probabilities_v==0;
    num_zeros=sum(fl0_v);
    corrected_probabilities_v = probabilities_v;
    if num_zeros>0
        corrected_probabilities_v(fl0_v)=epsi;

 corrected_probabilities_v(~fl0_v)=corrected_probabilities_v(~fl0_v)*(1-
num_zeros*epsi);
    end
    flagstop=0;
    str='';
end
```

Published with MATLAB® R2018b

```
function [K,ki_est,D_alpha,P_val,flag_ski,flagstop,str]...
    =estim_K(kidv,kiuv,sl_alpha,flagmethod,N,flagplot,flag_lang)
% ESTIM_K executes the uniform method to find point estimates of
 interval
%   elicited scaling constants of a n-D utility function
% ESTIM_K performs:
%       a) Monte Carlo estimation or analytical appoximation of the
 PDF
%           of the scaling constants sum;
%       b) a two-tail test to check if the sum of the scaling
 constants of
%           a n-D utility function is equal, greater or smaller than
 one;
%       c) calculation of the scaling constants point estimates in the
 cases
%            when their sum is equal, greater or smaller than one;
%       d) calculation of the general scaling constant k in the cases
%            when the scaling constants sum is equal, greater or
 smaller
%            than one;
%
% [K,ki_est,D_alpha,P_val,flag_ski,flagstop,str]...
%
 =estim_K(kidv,kiuv,sl_alpha,flagmethod,N,flagplot,flag_lang)
%
% kidv          i-dimensional column vector, containing the lower
%               boundaries of the scaling constants
% kiuv          i-dimensional column vector, containing the upper
%               boundaries of the scaling constants
% sl_alpha      the significance level of the two-tail test with H0
 being that
%               the scaling constants sum to one, and H1 being that
 the
%               scaling constants do not sum to one. By default,
%               sl_alpha=0.05
% flagmethod    flag for the method of estimation of the PDF of the
 scaling
%               constants sum:
%                   flagmethod=1 for Monte Carlo;
%                   flagmethod=2 for analytical approximation.
%               By default, flagmethod=2
% N             half of the synthetic data points to build the PDF of
 the
%               scaling constant sum if flagmethod=1, or number of the
 data
%               points for the analytical approximation of the PDF
%               of the scaling constants sum. By default, N=3000
% flagplot      flag for plot when flagplot==1
% flag_lang     flag for text language:
%               if flag_lang=1, all text prints in Bulgarian.
%               if flag_lang=0, all text prints in English.
%               By default, flag_lang=0
```

```
%
% K             the value of the general scaling constant k,
% ki_est        i-dimensional column vector of the resulting point
%               estimates of the scaling constants, obtained as
%               ki_est=kidv+(kiuv-kidv)*D_alpha, other
% D_alpha       the correction coefficient to transform uncertainty
 intervals
%               of scaling constants into point estimates. D_alpha is
 0.5
%               unless H0 is not rejected, when D_alpha=(1-skid)/
(skiu-skid),
%               where skid and skiu are respectively the sums of kidv
 and kiuv
% P_val         the p-value of the test, representing the probalility
 of
%               observing a non-unit sum of the scaling constants,
 when H0 is true
% flag_ski      sting flag indicating whether the sum of scaling
 constants
%               exceeds, is smaller or equal to one
% flagstop      flag for abnormal termination of 'estim_K' if
%               flagstop is not 0
% str           text string containing message from 'estim_K' if
%               flagstop is not 0

    estim_K_ic_handle=@estim_K_ic;
    % input checks
    er=0;
    flag_font='Times New Roman Cyr';
    flag_size=10;
    if nargin==6
        flag_lang=0;
    elseif nargin==5
        flag_lang=0;
        flagplot=0;
    elseif nargin==4 && flagmethod==2
        flag_lang=0;
        flagplot=0;
        N=101;
    elseif nargin==4 && flagmethod==1
        flag_lang=0;
        flagplot=0;
        N=3000;
    elseif nargin==3
        flag_lang=0;
        flagplot=0;
        flagmethod=2;
        N=101;
    elseif nargin==2
        flag_lang=0;
        flagplot=0;
        flagmethod=2;
        N=101;
        sl_alpha=.05;
```

```
    elseif nargin<2
        flagstop=1;
        str='Incorrect number of input parameters (message from
''estim_K'')';
        [K,ki_est,D_alpha,P_val]=deal([]);
        flag_ski='';
        disp(str);
        return;
    end
    if er==0

 [kidv,kiuv,sl_alpha,flagmethod,N,flagplot,flag_lang,flagstop,str,er]...

 =feval(estim_K_ic_handle,kidv,kiuv,sl_alpha,flagmethod,N,flagplot,flag_lang);
    end
    if flagstop~=0
        [K,ki_est,D_alpha,P_val]=deal([]);
        flag_ski='';
        fprintf(1,'%s\ninternal error code er=%i\n',str,er);
        return
    end
    % calculation
    funct_q_handle=@funct_q;
    estim_PDF_anal_handle=@estim_PDF_anal;
    emplawid_handle=@emplawid;
    num_sc=size(kidv,1);
    skid=sum(kidv);
    skiu=sum(kiuv);
    if flagmethod==1 && (flagplot==1 || (skid<1 && skiu>1))
        rand_sc=kidv*ones(1,N)+((kiuv-
kidv)*ones(1,N)).*rand(num_sc,N);
        rand_sum=sum(rand_sc);
        rand_sum=[rand_sum skid+skiu-rand_sum];
        N=2*N;
        b=floor(sqrt(N)/2);

 [~,~,xf_plot,PDFx_plot,xF,CDFx]=feval(emplawid_handle,rand_sum',b,0,skid,skiu);
        if 1<=xF(1)
            P_val=0;
        elseif xF(end)<=1
            P_val=1;
        else
            P_val=interp1(xF,CDFx,1);
        end
    elseif flagmethod==2 && (flagplot==1 || (skid<1 && skiu>1))
        [~,indsort]=sort(kiuv-kidv,1,'descend');
        kidvs=kidv(indsort);
        kiuvs=kiuv(indsort);
        if kidvs(1)>=kiuvs(1)
            x=[skid-eps,skid,skid+eps];
            fx=[0 1/eps 0];
        elseif kidvs(1)<kiuvs(1)
            x=[kidvs(1),kiuvs(1)];
            fx=ones(1,2)./(x(2)-x(1));
```

```
            for i=2:num_sc
                if kidvs(i)>=kiuvs(i)
                    x=x+kidvs(i);
                elseif kidvs(i)<kiuvs(i)
                    if i==2
                        x2=min([x(1)+kiuvs(2),x(2)+kidvs(2)]);
                        x3=max([x(1)+kiuvs(2),x(2)+kidvs(2)]);
                        fx23=2/(x(2)-x(1)+kiuvs(2)-kidvs(2)+x3-x2);
                        if x2<x3
                            x=[x(1)+kidvs(2) x2 x3 x(2)+kiuvs(2)];
                            fx=[0 fx23 fx23 0];
                        else
                            x=[x(1)+kidvs(2) (x2+x3)/2 x(2)+kiuvs(2)];
                            fx=[0 fx23 0];
                        end
                    elseif i>2

 [x,fx]=feval(estim_PDF_anal_handle,x,fx,kidvs(i),kiuvs(i),N);
                    end
                end
            end
        end
        if skid<1 && skiu>1
            fx1=interp1(x,fx,1);
            M=sum(x<1);
            P_val=(fx1+fx(M))*(1-x(M))/2;
            if M>1
                if length(x)>4
                    P_val=P_val+(x(M)-x(1))/(M-1)*(2*sum(fx(2:
(M-1)))+fx(1)+fx(M))/2;
                else
                    for i=1:(M-1)
                        P_val=P_val+(x(i+1)-x(i))*(fx(i+1)+fx(i))/2;
                    end
                end
            end
        else
            P_val=0;
        end
    end
    if flagplot==1 || (skid<1 && skiu>1)
        if P_val<0.5
            P_val=2*P_val;
            if P_val<sl_alpha
                flag_ski='more';
            else
                flag_ski='exact';
            end
        else
            P_val=2*(1-P_val);
            if P_val<sl_alpha
                flag_ski='less';
            else
                flag_ski='exact';
```

```
        end
    end
end
if skid==1 && skiu==1
    P_val=1;
    flag_ski='exact';
elseif skid>=1
    P_val=0;
    flag_ski='more';
elseif skiu<=1
    P_val=0;
    flag_ski='less';
end
switch flag_ski
    case 'more'
        D_alpha=0.5;
        ki_est=(kidv+kiuv)./2;
        qu=-eps*10;
        while 1
            fqu=feval(funct_q_handle,qu,ki_est);
            if fqu<0
                error('you are not supposed to be here');
            elseif fqu==0
                qu=2*qu;
            elseif fqu>0
                break
            end
        end
        K=fzero(funct_q_handle,[-1;qu],[],ki_est);
    case 'less'
        D_alpha=0.5;
        ki_est=(kidv+kiuv)./2;
        qd=eps*10;
        while 1==1
            fqd=feval(funct_q_handle,qd,ki_est);
            if fqd<0
                error('you are not supposed to be here');
            elseif fqd==0
                qd=2*qd;
            elseif fqd>0
                break;
            end
        end
        qu=1;
        while 1==1
            fqu=feval(funct_q_handle,qu,ki_est);
            if fqu<0
                K=fzero(funct_q_handle,[qd,qu],[],ki_est);
                break;
            elseif fqu>0
                qd=qu;
                qu=2*qu;
            elseif fqu==0
                K=qu;
```

```
                    break
                end
            end
        case 'exact'
            if skiu>skid
                D_alpha=(1-skid)/(skiu-skid);
            else
                D_alpha=0.5;
            end
            ki_est=kidv+(kiuv-kidv)*D_alpha;
            K=0;
    end
    if flagplot==1
        figure(1);
        close(1);
        figure(1);
        set(gca,'FontName',flag_font,'FontSize',flag_size);
        hold on;
        if flagmethod==1
            plot(xf_plot,PDFx_plot,'k-');
        elseif flagmethod==2
            plot(x,fx,'k-');
        end
        ax=axis;
        delx=0.05*(ax(2)-ax(1));
        dely=0.05*(ax(4)-ax(3));
        axis(ax+[-delx delx 0 dely]);
        if flag_lang==0
            xlabel('sum of the scaling constants, \Sigmak_i');
        elseif flag_lang==1
            xlabel('ñóìà îò ñêàëèðàùèõ êîíñòàíòè, \it\Sigmak_i');
        end
        ylabel('\itPDF(\it\Sigmak_i)');
        switch flag_ski
            case 'more'
                if flag_lang==0
                    title({'The sum of scaling constants is greater
 than 1'...
                        ;sprintf('p-value= %1.2e; significance level
 %1.2e',P_val,sl_alpha)});
                elseif flag_lang==1
                    title({'Ñóìàòà îò ñêàëèðàùèõ êîíñòàíòè å ïî-
ãîëÿìà îò 1'...
                        ;sprintf('p_{value}= %1.2e; íèâî íà çíà÷èìîñò
 %1.2e',P_val,sl_alpha)});
                end
            case 'less'
                if flag_lang==0
                    title({'The sum of scaling constants is less than
 1'...
                        ;sprintf('p-value= %1.2e; significance level
 %1.2e',P_val,sl_alpha)});
                elseif flag_lang==1
```

```
                    title({'Ñóìàòà íà ñêàëèðàùèõ êîíñòàíòè å ïî-ìàëêà
îò 1'...
                        ;sprintf('p_{value}= %1.2e; íèâî íà çíà÷èìîñò
%1.2e',P_val,sl_alpha)},'FontName','Times');
                end
            case 'exact'
                if flag_lang==0
                    title({'The sum of scaling constants is not
rejected to be 1'...
                        ;sprintf('p-value= %1.2e; significance level
%1.2e',P_val,sl_alpha)});
                elseif flag_lang==1
                    title({'Äâóíè÷íîñòà íà ñóìàòà íà ñêàëèðàùèõ
êîíñòàíòè íå ìîæå äà ñå îòõâúðëè'...
                        ;sprintf('p_{value}= %1.2e; íèâî íà çíà÷èìîñò
%1.2e',P_val,sl_alpha)},'FontName','Times');
                end
        end
        if K==0
            qd=-1;
            qu=1;
        elseif K<0
            qd=-1;
            qu=0;
        elseif K>0
            qd=0;
            qu=2*K;
        end
        q=linspace(qd,qu,100);
        fq=q;
        for i=1:length(q)
            fq(i)=feval(funct_q_handle,q(i),ki_est);
        end
        figure(2);
        close(2);
        figure(2);
        set(gca,'FontName',flag_font,'FontSize',flag_size);
        hold on;
        plot(q,fq);
        ax=axis;
        dely=0.05*(ax(4)-ax(3));
        axis(ax+[0 0 -dely dely]);
        ax=axis;
        hold on
        plot([ax(1) K K],[0 0 ax(3)],':');
        xlabel('\itq');
        ylabel('\itf\rm(\itq\rm)');
        if flag_lang==0
            title(sprintf('scaling constant K=%1.3e',K));
        elseif flag_lang==1
            title(sprintf('îáùà ñêàëèðàùà êîíñòàíòà K=%1.3e',K));
        end
    end
    flagstop=0;
```

```
    str='';
end


function fq=funct_q(q,ki_est)
    fq=1+q-prod(q*ki_est+1);
end


function [x,fx,flagstop,str]=estim_PDF_anal(y,fy,c,d,n)
% ESTIM_PDF_ANAL performs analytical approximation of the PDF of
% sum of two variables, the first with arbitrary distribution and the
% second with uniform distribution
%
% [x,fx,flagstop,str]=estim_PDF_anal(y,fy,c,d,n)
%
% y             row-vector of data points of the PDF of the first
 variable
% fy            row-vector of PDF values for the data points in y
% c             the lower boundary of the uniform distribution of the
%               second variable
% d             the upper boundary of the uniform distribution of the
%               second variable
% n             number of the data points for the analytical
 approximation
%               of PDF for the sum of the two variables. By default,
 n=101.
%
% x             row-vector of data points of the PDF for the sum of
 the two
%               variables
% fx            the PDF values for the data points in x
% flagstop      flag for abnormal termination of 'estim_PDF_anal' if
%               flagstop is not 0
% str           text string containing message from 'estim_PDF_anal'
 if
%               flagstop is not 0
    er=0;
    estim_PDF_anal_ic_handle=@estim_PDF_anal_ic;
    % input checks
    if nargin==4
        n=101;
    elseif nargin<4
        flagstop=1;
        str='Incorrect number of input parameters (message from
 ''estim_PDF_anal'')';
        [x,fx]=deal([]);
        return
    end
    if er==0

 [y,fy,c,d,n,flagstop,str,er]=feval(estim_PDF_anal_ic_handle,y,fy,c,d,n);
    end
    if flagstop~=0
```

```
        [x,fx]=deal([]);
        fprintf(1,'%s\ninternal error code er=%i\n',str,er);
        return
    end
    % calculation
    a=y(1);
    b=y(end);
    x=linspace(a+c,b+d,n);
    fix=x;
    fix([1,end])=0;
    yd=max(a,x'-d);
    yu=min(b,x'-c);
    for i=2:(n-1);
        yt=[yd(i) y(y>yd(i) & y<yu(i)) yu(i)];
        fyt=interp1(y,fy,yt);
        fix(i)=sum((yt(2:end)-yt(1:(end-1))).*(fyt(2:end)+fyt(1:
(end-1))))/2;
    end
    fx=fix/sum((x(2:end)-x(1:(end-1))).*(fix(2:end)+fix(1:
(end-1))))*2;
    flagstop=0;
    str='';
end

function [y,fy,c,d,n,flagstop,str,er]=estim_PDF_anal_ic(y,fy,c,d,n)
    flagstop=0;
    str='';
    er=0;
    if ~ismatrix(y)
        er=1;
    end
    if er==0 && size(y,1)~=1
        er=2;
    end
    if er~=0
        flagstop=1;
        str='Incorrect input parameter y (message from
 ''estim_PDF_anal'')';
        [y,fy,c,d,n]=deal([]);
        return
    end
    er=chcknmat(fy,1,size(y,2));
    if er==0 && any(fy<0)
        er=1;
    end
    if er~=0
        flagstop=2;
        str='Incorrect input parameter fy (message from
 ''estim_PDF_anal'')';
        [y,fy,c,d,n]=deal([]);
        return
    end
    er=chcknmat(c,1,1);
    if er~=0
```

```
        flagstop=3;
        str='Incorrect input parameter c (message from
 ''estim_PDF_anal'')';
        [y,fy,c,d,n]=deal([]);
        return
    end
    er=chcknmat(d,1,1);
    if er==0 && d<=c
        er=1;
    end
    if er~=0
        flagstop=4;
        str='Incorrect input parameter d (message from
 ''estim_PDF_anal'')';
        [y,fy,c,d,n]=deal([]);
        return
    end
    [er,n]=chcknint(n,30,101);
    if er~=0
        flagstop=5;
        str='Incorrect input parameter n (message from
 ''estim_PDF_anal'')';
        [y,fy,c,d,n]=deal([]);
        return
    end
end

function
 [kidv,kiuv,sl_alpha,flagmethod,N,flagplot,flag_lang,flagstop,str,er]...
    =estim_K_ic(kidv,kiuv,sl_alpha,flagmethod,N,flagplot,flag_lang)
    flagstop=0;
    str='';
    er=0;
    if ~ismatrix(kidv)
        er=1;
    end
    if er==0 && size(kidv,2)~=1
        kidv=kidv';
    end
    if er==0
        if size(kidv,2)~=1
            er=2;
        else
            if size(kidv,1)<2
                er=3;
            end
        end
    end
    if er==0
        er=chcknmat(kidv,size(kidv,1),1);
    end
    if er==0
        if any(kidv<0)
            er=4;
```

```
        elseif any(kidv>1)
            er=5;
        end
    end
    if er~=0
        flagstop=1;
        str='Incorrect input parameter kidv (message from
''estim_K'')';
        [kidv,kiuv,sl_alpha,flagmethod,N,flagplot,flag_lang]=deal([]);
        return
    end
    er=0;
    if ~ismatrix(kiuv)
        er=1;
    elseif size(kiuv,2)~=1
        kiuv=kiuv';
    end
    if er==0
        er=chcknmat(kiuv,size(kidv,1),1);
    end
    if er==0
        if any(kiuv>=1)
            er=3;
        elseif any(kidv>kiuv)
            er=4;
        end
    end
    if er~=0
        flagstop=2;
        str='Incorrect input parameter kiuv (message from
''estim_K'')';
        [kidv,kiuv,sl_alpha,flagmethod,N,flagplot,flag_lang]=deal([]);
        return
    end
    [er,sl_alpha]=chcknmat(sl_alpha,1,1,.05);
    if er==0 && (sl_alpha>=1 || sl_alpha<=0)
        er=1;
    end
    if er~=0
        flagstop=3;
        str='Incorrect input parameter sl_alpha (message from
''estim_K'')';
        [kidv,kiuv,sl_alpha,flagmethod,N,flagplot,flag_lang]=deal([]);
        return
    end
    [er,flagmethod]=chcknint(flagmethod,[1 2],2);
    if er~=0
        flagstop=4;
        str='Incorrect input parameter flagmethod (message from
''estim_K'')';
        [kidv,kiuv,sl_alpha,flagmethod,N,flagplot,flag_lang]=deal([]);
        return
    end
    er=0;
```

```
    if flagmethod==1
        [er,N]=chcknint(N,30,3000);
    elseif flagmethod==2
        [er,N]=chcknint(N,30,101);
    end
    if er~=0
        flagstop=5;
        str='Incorrect input parameter N (message from ''estim_K'')';
        [kidv,kiuv,sl_alpha,flagmethod,N,flagplot,flag_lang]=deal([]);
        return
    end
    [er,flagplot]=chcknint(flagplot,[0 1],0);
    if er~=0
        flagstop=6;
        str='Incorrect input parameter flagplot (message from
 ''estim_K'')';
        [kidv,kiuv,sl_alpha,flagmethod,N,flagplot,flag_lang]=deal([]);
        return
    end
    [er,flag_lang]=chcknint(flag_lang,[0 1],0);
    if er~=0
        flagstop=7;
        str='Incorrect input parameter flag_lang (message from
 ''estim_K'')';
        [kidv,kiuv,sl_alpha,flagmethod,N,flagplot,flag_lang]=deal([]);
        return
    end
end
```

Published with MATLAB® R2018b

```
function [ multi_attribute_utility ] =
 evaluate_MAU_pump(attribute_values_v)
% evaluate_MAU_pump translates the given attribute values vector into
 a single value of multi-attribute utility
% It uses values specified regarding pump maintenance

% Input:  attribute_values_v        1x6 vector where each value
 corresponds to an
%                                   attribute value of a multi-
attribute
%                                   consequence double
% Output: multi-attribute_utility   A single output value, assuming a
%                                   multiplicative combination of 6
%                                   attribute function evaluations
 double
%Parameters of the attribute functions are set within the
 utils_rng_database.m function, as
%well as their weights. Attribute functions are fitted as ARCTAN

%CALCULATIONS:
    %Load utility values and ki_weight ranges for pump
    inp_filename1_str = ['utils' '\' 'survey_utils'];
    load(inp_filename1_str,'ki_weights_m');
    %Calculate ki point estimates and overall k value
    [K,ki_est_v,~,P_val,~,~,~]...
        =estim_K(ki_weights_m(:,1),ki_weights_m(:,2)); %#ok<ASGLU>
    %Create storage vector
    weighted_utilities_v = zeros(length(attribute_values_v),1);
    % Evaluate utility function for each attribute and multiply by its
 weight
    %1. Downtime using attribute_values_v(1)
    %This is an ARCTAN function
    % Load data
    inp_filename1 = ['utils' '\' 'pump_attr_funct_1_params'];
    load(inp_filename1,'aopt','x0opt','xd','xu');
    a1 = aopt;          b1 = x0opt;
    xu1 = xu;           xd1 = xd;
    weighted_utilities_v(1) =
 ki_est_v(1)*(arctan_function(a1,b1,attribute_values_v(1),xu1,xd1));
    clear aopt x0opt xd xu
    %2. Replacement cost of the pump using attribute_values_v(2)
    %This is an ARCTAN function
    inp_filename2 = ['utils' '\' 'pump_attr_funct_2_params'];
    load(inp_filename2,'aopt','x0opt','xd','xu');
    a2 = aopt;           b2 = x0opt;
    xu2 = xu;            xd2 = xd;
    weighted_utilities_v(2) =
 ki_est_v(2)*(arctan_function(a2,b2,attribute_values_v(2),xu2,xd2));
    clear aopt x0opt xd xu
    %3. Replacement cost of the ship using attribute_values_v(3)
    %%This is an ARCTAN function
    inp_filename3 = ['utils' '\' 'pump_attr_funct_3_params'];
```

```
    load(inp_filename3,'aopt','x0opt','xd','xu');
    a3 = aopt;          b3 = x0opt;
    xu3 = xu;         xd3 = xd;
    weighted_utilities_v(3) =
 ki_est_v(3)*(arctan_function(a3,b3,attribute_values_v(3),xu3,xd3));
    clear aopt x0opt xd xu
    %4. Expected number of people injured using attribute_values_v(4);
    %%This is an ARCTAN function
    inp_filename4 = ['utils' '\' 'pump_attr_funct_4_params'];
    load(inp_filename4,'aopt','x0opt','xd','xu');
    a4 = aopt;          b4 = x0opt;
    xu4 = xu;           xd4 = xd;
    weighted_utilities_v(4) =
 ki_est_v(4)*(arctan_function(a4,b4,attribute_values_v(4),xu4,xd4));
    clear aopt x0opt xd xu
    %5. Routine maintenance cost using attribute_values_v(5)
    %Maintenance cost is an ARCTAN function
    inp_filename4 = ['utils' '\' 'pump_attr_funct_5_params'];
    load(inp_filename4,'aopt','x0opt','xd','xu');
    a5 = aopt;    b5 = x0opt;
    xu5 = xu;      xd5 = xd;
    weighted_utilities_v(5) =
 ki_est_v(5)*(arctan_function(a5,b5,attribute_values_v(5),xu5,xd5));
    clear aopt x0opt xd xu
    %6. Contractual compliance using attribute_values_v(6);
    %This is a binary function, so we multiply the value by the weight
 only
    weighted_utilities_v(6) = ki_est_v(6)*attribute_values_v(6);
    % Compute the MAU value using a multiplicative combination and
 overall constant K determined from estim_k.m
    % Calculate the multi attribute utility
    RHS = prod(1+(K.*weighted_utilities_v)); %Refer to Clemen(1996),
 pg. 591 for equation
    multi_attribute_utility = (RHS - 1)/K;
end
```

Published with MATLAB® R2018b

```
function [ multi_attribute_utility ] =
 evaluate_MAU_test(attribute_values_v)
%evaluate_MAU_pump translates the given attribute values vector into a
 single value of multi-attribute utility
% It uses test values
%
% Input:  attribute_values_v        1x6 vector where each value
 corresponds to an
%                                   attribute value of a multi-
attribute
%                                   consequence
% Output: multi-attribute_utility   A single output value, assuming a
%                                   multiplicative combination of 6
%                                   attribute function evaluations
%Parameters of the attribute functions are set within the
 utils_rng_database.m function, as
%well as their weights. Attribute functions are fitted as ARCTAN

%CALCULATIONS:
    %Load utility values and ki_weight ranges for pump
    inp_filename1_str = ['utils' '\' 'test_utils'];
    load(inp_filename1_str,'ki_weights_m');
    %Calculate ki point estimates and overall k value
    [K,ki_est_v,~,P_val,~,~,~]...
        =estim_K(ki_weights_m(:,1),ki_weights_m(:,2)); %#ok<ASGLU>
    %Create storage vector
    weighted_utilities_v = zeros(length(attribute_values_v),1);
    % Evaluate utility function for each attribute and multiply by its
 weight
    %1. Downtime using attribute_values_v(1)
    %This is an ARCTAN function
    % Load data
    inp_filename1 = ['utils' '\' 'test_attr_funct_1_params'];
    load(inp_filename1,'aopt','x0opt','xd','xu');
    a1 = aopt;          b1 = x0opt;
    xu1 = xu;           xd1 = xd;
    weighted_utilities_v(1) =
 ki_est_v(1)*(arctan_function(a1,b1,attribute_values_v(1),xu1,xd1));
    clear aopt x0opt xd xu
    %2. Replacement cost of the pump using attribute_values_v(2)
    %This is an ARCTAN function
    inp_filename2 = ['utils' '\' 'test_attr_funct_2_params'];
    load(inp_filename2,'aopt','x0opt','xd','xu');
    a2 = aopt;           b2 = x0opt;
    xu2 = xu;            xd2 = xd;
    weighted_utilities_v(2) =
 ki_est_v(2)*(arctan_function(a2,b2,attribute_values_v(2),xu2,xd2));
    clear aopt x0opt xd xu
    %3. Replacement cost of the ship using attribute_values_v(3)
    %This is an ARCTAN function
    inp_filename3 = ['utils' '\' 'test_attr_funct_3_params'];
    load(inp_filename3,'aopt','x0opt','xd','xu');
```

```
    a3 = aopt;          b3 = x0opt;
    xu3 = xu;         xd3 = xd;
    weighted_utilities_v(3) =
 ki_est_v(3)*(arctan_function(a3,b3,attribute_values_v(3),xu3,xd3));
    clear aopt x0opt xd xu
    %4. Expected number of people injured using attribute_values_v(4);
    %This is an ARCTAN function
    inp_filename4 = ['utils' '\' 'test_attr_funct_4_params'];
    load(inp_filename4,'aopt','x0opt','xd','xu');
    a4 = aopt;          b4 = x0opt;
    xu4 = xu;           xd4 = xd;
    weighted_utilities_v(4) =
 ki_est_v(4)*(arctan_function(a4,b4,attribute_values_v(4),xu4,xd4));
    clear aopt x0opt xd xu
    %5. Routine maintenance cost using attribute_values_v(5)
    %Maintenance cost is an ARCTAN function
    inp_filename4 = ['utils' '\' 'test_attr_funct_5_params'];
    load(inp_filename4,'aopt','x0opt','xd','xu');
    a5 = aopt;    b5 = x0opt;
    xu5 = xu;      xd5 = xd;
    weighted_utilities_v(5) =
 ki_est_v(5)*(arctan_function(a5,b5,attribute_values_v(5),xu5,xd5));
    clear aopt x0opt xd xu
    %6. Contractual compliance using attribute_values_v(6);
    %This is a binary function, so we multiply the value by the weight
 only
    weighted_utilities_v(6) = ki_est_v(6)*attribute_values_v(6);
    % Compute the MAU value using a multiplicative combination and
 overall constant K determined from estim_k.m
    % Calculate the multi attribute utility
    RHS = prod(1+(K.*weighted_utilities_v)); %Refer to Clemen(1996),
 pg. 591 for equation
    multi_attribute_utility = (RHS - 1)/K;
end
```

Published with MATLAB® R2018b

```
function [exp_util_v] = eval_tree_atsea_emergency_pump(Pos_v,flag_dat)
% eval_tree_atsea_pump evaluates a decision tree using Pos_v
 posterior ...
% probabilities to produce expected utilities
% We consider that an emergency is occurring in this case so we can't
 stop
% the pump eg. it is being used to put out a fire
%
% Inputs:  Pos_v        A row vector of 8 posterior probabilities
%          flag_dat     Integer equalling 4 used to select which
 dataset
%                       was used to produce posterior probabilities
 Pos_v
% Output:  exp_util_v   A column vector of expected utilities
 evaluated using
%                       all five lottery models

% CALCULATIONS:
    %Load subjective probability data
    inp_filename1_str=['DATA'
 int2str(flag_dat) '\subjective_probability'];
    load(inp_filename1_str,'subjective_probability_worstcase_m');
    %Calculate multi-attribute utility from Pos_v
    [multi_attribute_utilities_m] =
 multi_attribute_utilities_atsea_pump(Pos_v);
    %Calculate expected utilities using lottery models
    exp_util_v =zeros(5,1);
    exp_util_v(1)= 0;
    exp_util_v(2)=recursive_ICOL(Pos_v,...
        subjective_probability_worstcase_m(2,:),...
        multi_attribute_utilities_m(2,1:8));
    exp_util_v(3)=recursive_ICOL(Pos_v,...
        subjective_probability_worstcase_m(3,:),...
        multi_attribute_utilities_m(3,1:8));
    exp_util_v(4)=recursive_ICOL(Pos_v,...
        subjective_probability_worstcase_m(4,:),...
        multi_attribute_utilities_m(4,1:8));
    exp_util_v(5)=recursive_ICOL(Pos_v,...
        subjective_probability_worstcase_m(5,:),...
        multi_attribute_utilities_m(5,1:8));
end
```

Published with MATLAB® R2018b

```
function [exp_util_v] = eval_tree_atsea_emergency_test(Pos_v,flag_dat)
% eval_tree_atsea_emergency_test evaluates a decision tree using Pos_v
 posterior probabilities
% We consider that an emergency is occurring in this case so we can't
 stop
% the pump eg. it is being used to put out a fire
%
% Inputs:  Pos_v        A row vector of 8 posterior probabilities
%          flag_dat     Integer equalling 4 used to select which
 dataset
%                       was used to produce posterior probabilities
 Pos_v
% Output:  exp_util_v   A column vector of expected utilities
 evaluated using
%                       all five lottery models

% CALCULATIONS:
    %Load subjective probability data
    inp_filename1_str=['DATA'
 int2str(flag_dat) '\subjective_probability_testdata'];

 load(inp_filename1_str,'subjective_probability_worstcase_testdata_m');
    %Calculate multi-attribute utility from Pos_v
    [multi_attribute_utilities_m] =
 multi_attribute_utilities_atsea_test(Pos_v);
    %Calculate expected utilities using lottery models
    exp_util_v =zeros(5,1);
    exp_util_v(1)= 0;
    exp_util_v(2)=recursive_ICOL(Pos_v,...
        subjective_probability_worstcase_testdata_m(2,:),...
        multi_attribute_utilities_m(2,1:8));
    exp_util_v(3)=recursive_ICOL(Pos_v,...
        subjective_probability_worstcase_testdata_m(3,:),...
        multi_attribute_utilities_m(3,1:8));
    exp_util_v(4)=recursive_ICOL(Pos_v,...
        subjective_probability_worstcase_testdata_m(4,:),...
        multi_attribute_utilities_m(4,1:8));
    exp_util_v(5)=recursive_ICOL(Pos_v,...
        subjective_probability_worstcase_testdata_m(5,:),...
        multi_attribute_utilities_m(5,1:8));
end
```

Published with MATLAB® R2018b

```
function [exp_util_v] = eval_tree_atsea_pump(Pos_v,flag_dat)
% eval_tree_atsea_pump evaluates a decision tree using Pos_v posterior
 probabilities
%
% Inputs:  Pos_v        A row vector of 8 posterior probabilities
%          flag_dat     Integer equalling 3 used to select which
 dataset
%                       was used to produce posterior probabilities
 Pos_v
% Output:  exp_util_v   A column vector of expected utilities
 evaluated using
%                       all five lottery models

% CALCULATIONS:
    %Load subjective probability data
    inp_filename1_str=['DATA'
 int2str(flag_dat) '\subjective_probability'];
    load(inp_filename1_str,'subjective_probability_worstcase_m');
    %Calculate multi-attribute utility from Pos_v
    [multi_attribute_utilities_m] =
 multi_attribute_utilities_atsea_pump(Pos_v);
    %Calculate expected utilities using lottery models
    exp_util_v =zeros(5,1);

 exp_util_v(1)=simple_lottery(Pos_v,multi_attribute_utilities_m(1,:));
    exp_util_v(2)=recursive_ICOL(Pos_v,...
        subjective_probability_worstcase_m(2,:),...
        multi_attribute_utilities_m(2,1:8));
    exp_util_v(3)=recursive_ICOL(Pos_v,...
        subjective_probability_worstcase_m(3,:),...
        multi_attribute_utilities_m(3,1:8));
    exp_util_v(4)=recursive_ICOL(Pos_v,...
        subjective_probability_worstcase_m(4,:),...
        multi_attribute_utilities_m(4,1:8));
    exp_util_v(5)=recursive_ICOL(Pos_v,...
        subjective_probability_worstcase_m(5,:),...
        multi_attribute_utilities_m(5,1:8));
end
```

Published with MATLAB® R2018b

```
function [exp_util_v] = eval_tree_atsea_test(Pos_v,flag_dat)
%eval_tree_atsea_test evaluates a decision tree using Pos_v posterior
 probabilities
%
% Inputs:  Pos_v        A row vector of 8 posterior probabilities
%          flag_dat     Integer equalling 3 used to select which
 dataset
%                       was used to produce posterior probabilities
 Pos_v
% Output:  exp_util_v   A column vector of expected utilities
 evaluated using
%                       all five lottery models

% CALCULATIONS:
    %Load subjective probability data
    inp_filename1_str=['DATA'
 int2str(flag_dat) '\subjective_probability_testdata'];

 load(inp_filename1_str,'subjective_probability_worstcase_testdata_m');
    %Calculate multi-attribute utility from Pos_v
    [multi_attribute_utilities_m] =
 multi_attribute_utilities_atsea_test(Pos_v);
    %Calculate expected utilities using lottery models
    exp_util_v =zeros(5,1);

 exp_util_v(1)=simple_lottery(Pos_v,multi_attribute_utilities_m(1,:));
    exp_util_v(2)=recursive_ICOL(Pos_v,...
        subjective_probability_worstcase_testdata_m(2,:),...
        multi_attribute_utilities_m(2,1:8));
    exp_util_v(3)=recursive_ICOL(Pos_v,...
        subjective_probability_worstcase_testdata_m(3,:),...
        multi_attribute_utilities_m(3,1:8));
    exp_util_v(4)=recursive_ICOL(Pos_v,...
        subjective_probability_worstcase_testdata_m(4,:),...
        multi_attribute_utilities_m(4,1:8));
    exp_util_v(5)=recursive_ICOL(Pos_v,...
        subjective_probability_worstcase_testdata_m(5,:),...
        multi_attribute_utilities_m(5,1:8));
end
```

Published with MATLAB® R2018b

```
function [exp_util_v] =
 eval_tree_atwharf_enginesrunning_pump(Pos_v,flag_dat)
% eval_tree_atwharf_enginesrunning_pump evaluates a decision tree
 using Pos_v posterior probabilities
%
% Inputs:  Pos_v        A row vector of 8 posterior probabilities
%          flag_dat     Integer equalling 2 used to select which
 dataset
%                       was used to produce posterior probabilities
 Pos_v
% Output:  exp_util_v   A column vector of expected utilities
 evaluated using
%                       all five lottery models

% CALCULATIONS:
    %Load subjective probabilities from file
    inp_filename_str = ['DATA'
 int2str(flag_dat) '\subjective_probability'];
    load(inp_filename_str,'subjective_probability_bestcase_m',...
        'subjective_probability_worstcase_m');
    %Calculate multi-attribute utility from Pos_v
    [multi_attribute_utilities_m] =
 multi_attribute_utilities_atwharf_pump(Pos_v);
    %Calculate expected utilities using lottery models
    exp_util_v = zeros(5,1);
    exp_util_v(1)=compound_lottery_A1(Pos_v,...
        subjective_probability_bestcase_m(1,:),...
        multi_attribute_utilities_m(1,:));
    exp_util_v(2)=recursive_ICOL(Pos_v,...
        subjective_probability_worstcase_m(2,:),...
        multi_attribute_utilities_m(2,1:8));
    exp_util_v(3)=recursive_ICOL(Pos_v,...
        subjective_probability_worstcase_m(3,:),...
        multi_attribute_utilities_m(3,1:8));
    exp_util_v(4)=recursive_ICOL(Pos_v,...
        subjective_probability_worstcase_m(4,:),...
        multi_attribute_utilities_m(4,1:8));
    exp_util_v(5)=recursive_ICOL(Pos_v,...
        subjective_probability_worstcase_m(5,:),...
        multi_attribute_utilities_m(5,1:8));
end
```

Published with MATLAB® R2018b

```
function [exp_util_v] =
 eval_tree_atwharf_enginesrunning_test(Pos_v,flag_dat)
% eval_tree_atwharf_enginesrunning_test evaluates a decision tree
 using Pos_v posterior probabilities
%
% Inputs:  Pos_v        A row vector of 8 posterior probabilities
%          flag_dat     Integer equalling 2 used to select which
 dataset
%                       was used to produce posterior probabilities
 Pos_v
% Output:  exp_util_v   A column vector of expected utilities
 evaluated using
%                       all five lottery models

% CALCULATIONS:
    %Load subjective probabilities from file
    inp_filename_str = ['DATA'
 int2str(flag_dat) '\subjective_probability_testdata'];

 load(inp_filename_str,'subjective_probability_bestcase_testdata_m',...
        'subjective_probability_worstcase_testdata_m');
    %Calculate multi-attribute utility from Pos_v
    [multi_attribute_utilities_m] =
 multi_attribute_utilities_atwharf_test(Pos_v);
    %Calculate expected utilities using lottery models
    exp_util_v = zeros(5,1);
    exp_util_v(1)=compound_lottery_A1(Pos_v,...
        subjective_probability_bestcase_testdata_m(1,:),...
        multi_attribute_utilities_m(1,:));
    exp_util_v(2)=recursive_ICOL(Pos_v,...
        subjective_probability_worstcase_testdata_m(2,:),...
        multi_attribute_utilities_m(2,1:8));
    exp_util_v(3)=recursive_ICOL(Pos_v,...
        subjective_probability_worstcase_testdata_m(3,:),...
        multi_attribute_utilities_m(3,1:8));
    exp_util_v(4)=recursive_ICOL(Pos_v,...
        subjective_probability_worstcase_testdata_m(4,:),...
        multi_attribute_utilities_m(4,1:8));
    exp_util_v(5)=recursive_ICOL(Pos_v,...
        subjective_probability_worstcase_testdata_m(5,:),...
        multi_attribute_utilities_m(5,1:8));
end
```

Published with MATLAB® R2018b

```
function [exp_util_v] =
 eval_tree_atwharf_norunning_pump(Pos_v,flag_dat)
% eval_tree_atwharf_norunning_pump evaluates a decision tree using
 Pos_v posterior probabilities
%
% Inputs:  Pos_v        A row vector of 8 posterior probabilities
%          flag_dat     Integer equalling 1 used to select which
 dataset
%                       was used to produce posterior probabilities
 Pos_v
% Output:  exp_util_v   A column vector of expected utilities
 evaluated using
%                       all five lottery models

% CALCULATIONS:
    % Load addittional subjective probabilities from file
    inp_filename_str = ['DATA'
 int2str(flag_dat) '\subjective_probability'];
    load(inp_filename_str,'subjective_probability_bestcase_m',...
        'subjective_probability_worstcase_m');
    % Calculate multi-attribute utility from Pos_v
    multi_attribute_utilities_m =
 multi_attribute_utilities_atwharf_pump(Pos_v);
    % Calculate expected utilities using lottery models
    exp_util_v = zeros(5,1);
    exp_util_v(1)=compound_lottery_A1(Pos_v,...
        subjective_probability_bestcase_m(1,:),...
        multi_attribute_utilities_m(1,:));
    exp_util_v(2)=recursive_ICOL(Pos_v,...
        subjective_probability_worstcase_m(2,:),...
        multi_attribute_utilities_m(2,1:8));
    exp_util_v(3)=recursive_ICOL(Pos_v,...
        subjective_probability_worstcase_m(3,:),...
        multi_attribute_utilities_m(3,1:8));
    exp_util_v(4)=recursive_ICOL(Pos_v,...
        subjective_probability_worstcase_m(4,:),...
        multi_attribute_utilities_m(4,1:8));
    exp_util_v(5)=recursive_ICOL(Pos_v,...
        subjective_probability_worstcase_m(5,:),...
        multi_attribute_utilities_m(5,1:8));
end
```

Published with MATLAB® R2018b

```
function [exp_util_v] =
 eval_tree_atwharf_norunning_test(Pos_v,flag_dat)
% eval_tree_atwharf_norunning_test evaluates a decision tree using
 Pos_v posterior probabilities
%
% Inputs:  Pos_v        A row vector of 8 posterior probabilities
%          flag_dat     Integer equalling 1 used to select which
 dataset
%                       was used to produce posterior probabilities
 Pos_v
% Output:  exp_util_v   A column vector of expected utilities
 evaluated using
%                       all five lottery models

% CALCULATIONS:
    % Load addittional subjective probabilities from file
    inp_filename_str = ['DATA'
 int2str(flag_dat) '\subjective_probability_testdata'];

 load(inp_filename_str,'subjective_probability_bestcase_testdata_m',...
        'subjective_probability_worstcase_testdata_m');
    % Calculate multi-attribute utility from Pos_v
    multi_attribute_utilities_m =
 multi_attribute_utilities_atwharf_test(Pos_v);
    % Calculate expected utilities using lottery models
    exp_util_v = zeros(5,1);
    exp_util_v(1)=compound_lottery_A1(Pos_v,...
        subjective_probability_bestcase_testdata_m(1,:),...
        multi_attribute_utilities_m(1,:));
    exp_util_v(2)=recursive_ICOL(Pos_v,...
        subjective_probability_worstcase_testdata_m(2,:),...
        multi_attribute_utilities_m(2,1:8));
    exp_util_v(3)=recursive_ICOL(Pos_v,...
        subjective_probability_worstcase_testdata_m(3,:),...
        multi_attribute_utilities_m(3,1:8));
    exp_util_v(4)=recursive_ICOL(Pos_v,...
        subjective_probability_worstcase_testdata_m(4,:),...
        multi_attribute_utilities_m(4,1:8));
    exp_util_v(5)=recursive_ICOL(Pos_v,...
        subjective_probability_worstcase_testdata_m(5,:),...
        multi_attribute_utilities_m(5,1:8));
end
```

Published with MATLAB® R2018b

```
function [x_noisy_c] = generate_noise(flag_dat,alf_proc)
%generate_noise produces vectors with levels of gaussian noise about
 the mean vector for the learning data sepcified by flag_dat
%   Inputs: flag_dat  An integer (1 ,2, 3 or 4) describing the
 decision
%                     context as follows:
%                     1 = Vessel is stationary at the wharf, no other
 machinery running
%                         in engine room
%                     2 = Vessel is stationary at the wharf, main
 engines are running
%                         in engine room
%                     3 = Vessel is slow steaming in the harbour and
 we can
%                         stop the pump if necessary
%                     4 = Vessel is slow steaming in the harbour and
 we cannot
%                         stop the pump because there is an emergency
 which it
%                         must be made available for
%          alf_proc   An integer specifying the noise level as a whole
 number
%                     > 0
%
%   Output: x_noisy_c A cell array containing all the noisy vectors
 produced
%                   within the function

%CALCULATIONS:
    %Load the learning data
    inp_filename1_str=['DATA' int2str(flag_dat) '\parameters'];
    load(inp_filename1_str,'Ls_c','WC_c','mu_c');
    %Determine classes
    c=length(Ls_c);
    dist_mean_v = mean_distance(mu_c);
    x_noisy_c = cell(c,1);
    for k=1:c
        Lsk_m=Ls_c{k};
        WCk_v=WC_c{k};
        Lsk_m=Lsk_m(:,WCk_v==1);
        nk=size(Lsk_m,2);
        x_noisy_m = zeros(size(Lsk_m,1),size(Lsk_m,2));
        for j=1:nk
            x_clean_v=Lsk_m(:,j);
            x_noisy_m(:,j)=
 normrnd(x_clean_v,alf_proc*dist_mean_v/100);
        end
        x_noisy_c{k} = x_noisy_m;
    end
end
```

Published with MATLAB® R2018b

```
function []=learning_database(selection)
% learning_database displays the learning samples which were used to
 develop the bayseian classifiers for
% decision analysis
% This function loads all learning samples and weighing coefficients
 stored within the system,
% enabling the user to view them only
%
%Input: selection  An integer (1 ,2, 3 or 4) describing the decision
%                  context as follows:
%                  1 = Vessel is stationary at the wharf, no other
 machinery running
%                      in engine room
%                  2 = Vessel is stationary at the wharf, main engines
 are running
%                      in engine room
%                  3 = Vessel is slow steaming in the harbour and we
 can
%                      stop the pump if necessary
%                  4 = Vessel is slow steaming in the harbour and we
 cannot
%                      stop the pump because there is an emergency
 which it
%                      must be made available for
%   Note on the datasets:
%            DATA(flag_dat)\learning_data_default.mat contains:
%                        class_names_c     8 x 1 cell matrix of class
 name strings
%                        features_c        85 x 1 cell matrix of
 feature name strings
%                        Ls_1_m ... Ls_8_m 85 x n learning sample
 matrices.
%                                          Ls_1_m has n = 50 while all
 others have n = 10
%                        WC1_v ... WC8_v_m 1 x n weighting coefficent
 matrices.
%                                          Weighting coefficents used
 for class 1
%

%CALCULATIONS:
    % Change selection into flag_dat and emergency variables
    flag_dat = selection;
    % String to describe data selected by user
    working_dataset_c ={'Vessel is at the wharf, no equipment
 running'...
        ,'Vessel is at the wharf, other machinery is running'...
        ,'Vessel is slow steaming in the harbour and it is not an
 emergency'...
        ,'Vessel is slow steaming in the harbour and it is an
 emergency'};
    choice = 0;
```

```
   while choice ~=3
       disp(strcat('Selected dataset
describes:',working_dataset_c{flag_dat}));
       choice = menu('Learning Database Options',...
           '1 = View Pump Learning Data',...
           '2 = Change datasets within Pump Learning Data',...
           '3 = Return to Main Menu');
       % Load and manage the data according to menu choices
       if (choice == 1) %View only
           view_again = 'Y';
           inp_filename1_str=['DATA'
int2str(flag_dat) '\learning_data'];
           load(inp_filename1_str,'class_names_c', 'features_c',...

 'Ls_1_m','Ls_2_m','Ls_3_m','Ls_4_m','Ls_5_m','Ls_6_m','Ls_7_m','Ls_8_m',...

 'WC1_v','WC2_v','WC3_v','WC4_v','WC5_v','WC6_v','WC7_v','WC8_v');
           learning_samples_c =
{Ls_1_m,Ls_2_m,Ls_3_m,Ls_4_m,Ls_5_m,Ls_6_m,Ls_7_m,Ls_8_m};
           w_coeff_c =
{WC1_v,WC2_v,WC3_v,WC4_v,WC5_v,WC6_v,WC7_v,WC8_v};
           total_samples =
size([Ls_1_m,Ls_2_m,Ls_3_m,Ls_4_m,Ls_5_m,Ls_6_m,Ls_7_m,Ls_8_m],2);
           features_c = features_c';
           while (view_again == 'Y') || (view_again == 'y')
               disp(strcat('Data loaded into workspace. Dataset
contains ...',num2str(length(class_names_c)),'... classes
and ...',...
                   num2str(total_samples),'... samples.'));
               view_class_data = menu('Please select a class to view
corresponding learning samples and weighting coefficents:',...
                   class_names_c);
               view_samples_m = learning_samples_c{view_class_data}';
               num_rows = size(view_samples_m,1);
               num_cols = size(view_samples_m,2);
               view_samples_t =
array2table(view_samples_m,'VariableNames',features_c);
               view_coeff_v = w_coeff_c{view_class_data}';
               view_coeff_t =
array2table(view_coeff_v,'VariableNames',{'Weighting_Coefficients'});
               disp(strcat('Displaying learning data for
class...',num2str(view_class_data),':',class_names_c{view_class_data}));
               disp(strcat('Dataset
contains ...',num2str(num_rows),'... learning samples
with ...',num2str(num_cols),'... features.'));
               disp(view_samples_t);
               disp(strcat('There is a corresponding vector
of ...',num2str(num_rows),'... weighting coefficients for this
class.'));
               disp(view_coeff_t);
               view_again = input('View another class dataset? Y/N as
character');
           end
       elseif (choice == 2)
```

```
            %Prompt user for a new choice
            selection = menu('Please select dataset of
 interest:','Vessel is at the wharf, no equipment running'...
                ,'Vessel is at the wharf, other machinery is
 running'...
                ,'Vessel is slow steaming in the harbour and it is not
 an emergency'...
                ,'Vessel is slow steaming in the harbour and it is an
 emergency');
            % Change selection into flag_dat and emergency variables
            flag_dat = selection;
        end
    end
    input('Returning to Main Menu. Press any key to continue.');
end
```

Published with MATLAB® R2018b

```
function [exit_flag] = main_menu()
%main_menu initialises the program to perform a maintenance decision
 analysis
%   In this case, the program relates to the maintenance of a
 shipboard
%   pump.
%   In this function, a menu is generated which specifies the options
%   available within the software and prompts the user to select them.
%
%   The user has the option to select one of multiple choices, with a
%   function running for each before ther function finishes and
 returns to
%   this menu or is stopped by the user and returns to this menu.
%
%    The key capability of this program is the decision analyisis of a
 pump
%    to determine maintenance actions for a given mixture of eight
 possible
%    states.
%    Bayesian linear discriminant classifiers were developed on
 supervised
%    data from the pump.
%    The probabilities output from the classification are used in a
%    decision tree to select the optimal policy.
%
%   Outputs:
%        exit_flag   An integer indicating program is running when set
%                    to zero, or 1 when user quits the program by
%                    selecting 'Q' or 'q' when prompted
%
%   Menu Options:
%                    1 = Calculate maintenance policy
%                    2 = Access Pump Measurement Database(View,
 Modify)
%                    3 = Access Learning Samples Database (View,
 Modify)
%                    4 = Access Utilities Database (View,Modify)
%                    5 = Access Subjective Probabilities Database
 (View, Modify)
%                    6 = Quit
%
%CALCULATIONS:
    % Clear matlab workspace
    clc;
    %Initalise variable to guide operation
    choice = 0;
    % Command line notifcation
    disp('Maintenance Decision Analysis of Shipboard Pump 2019 - Main
 Menu.');
    %Prepare menu text
    menu_header_str = 'Maintenance Decision Analysis of Shipboard Pump
 2019';
```

```
    while choice ~= 6
        choice = menu(menu_header_str,'1 = Calculate Maintenance
Policy',...
            '2 = Access Pump Measurement Database(View, Modify)',...
            '3 = Access Learning Samples Database (View)',...
            '4 = Access Utilities Database (View,Modify)',...
            '5 = Access Subjective Probabilities Database (View,
Modify)',...
            '6 = Quit');
        if choice == 1
        disp('Proceeding to calculation sub-menu ...');
        calculation_menu();
        elseif choice == 2
        disp('Proceeding to pump measurements database ...');
        measurement_database();
        elseif choice == 3
        disp('Proceeding to learning samples database ...');
        selection = menu('Please select dataset of interest:','Vessel
is at the wharf, no equipment running'...
        ,'Vessel is at the wharf, other machinery is running'...
        ,'Vessel is slow steaming in the harbour and it is not an
emergency'...
        ,'Vessel is slow steaming in the harbour and it is an
emergency');
        learning_database(selection);
        elseif choice == 4
        disp('Proceeding to multi-attribute utilities database ...');
        util_rng_database();
        elseif choice == 5
        disp('Proceeding to subjective probabilities database ...');
        selection = menu('Please select dataset of interest:','Vessel
is at the wharf, no equipment running'...
        ,'Vessel is at the wharf, other machinery is running'...
        ,'Vessel is slow steaming in the harbour and it is not an
emergency'...
        ,'Vessel is slow steaming in the harbour and it is an
emergency');
        subj_prob_database(selection);
        end
        disp('Closing program.');
        exit_flag = 1;
    end
end
```

Published with MATLAB® R2018b

```
function [dist_mean_v,absdist_m] = mean_distance(mu_c)
% mean_distance calculates the mean distance between the mean value
 vectors
%
%   Input: mu_c           A cell array containing vectors of mean
 values
%                         double
%
%   Outputs: dist_mean_v  A vector of mean values double
%            absdist_m    A matrix of absolute distance values of each
%                         value double from the relevant mean
%

% CALCULATIONS:
    c=length(mu_c);
    d=length(mu_c{1});
    mu_m=NaN(d,c);
    for k=1:c
        mu_m(:,k)=mu_c{k};
    end
    jmax=c*(c-1)/2;
    absdist_m=NaN(d,jmax);
    j=0;
    for k1=1:c
        for k2=(k1+1):c
            j=j+1;
            absdist_m(:,j)=abs(mu_m(:,k1)-mu_m(:,k2));
        end
    end
    dist_mean_v=sqrt(sum(absdist_m.^2,2)/jmax);
end
```

Published with MATLAB® R2018b

```
function []=measurement_database()
% measurement_database manages the pump measurements and test
 measurments for classification
%   This function loads all pump measurements stored within the
 system,
%   enabling the user to view and modify them
%   These data are unlabelled
%
%   Note on the datasets:
%            pumpdata_default.mat matrix of 85 features x 51 samples
 of
%            samples from a shipboard pump. This is not overwritten by
 the
%            software. For the physical meaning of these features and
 units of measurement, refer to
%            Thesis/LDA paper about the pump
%
%            test.mat is a randomly generated matrix of 85 features x
 100
%            samples which can be regenerated for testing purposes

% CALCULATIONS:
    choice = 0;
    while choice ~=9
        choice = menu('Measurement Database Options',...
            '1 = View Pump Data',...
            '2 = Modify Pump Data Values',...
            '3 = Add/remove Pump Data Measurements',...
            '4 = Restore Pump Data to Default',...
            '5 = View Test Data',...
            '6 = Modify Test Data Values',...
            '7 = Add/remove Test Data Measurements',...
            '8 = Restore Test Data to Default',...
            '9 = Return to Main Menu');
        % Load and manage the data according to menu choices
        if (choice == 1) %View only
            view_again = 'Y';
            inp_filename1_str=['meas','\','pumpdata'];
            load(inp_filename1_str,'pumpdata_m');
            while (view_again == 'Y') || (view_again == 'y')
                disp(strcat('Data loaded into workspace. Dataset size
 is ...',num2str(size(pumpdata_m,1)),'... features by ...',...
                    num2str(size(pumpdata_m,2)),'... samples.'));
                rows_v = input('Please input row numbers to view as a
 vector[a,b].');
                columns_v = input('Please input column numbers to view
 as a vector[a,b].');
                disp('Displaying specified range and pausing
 execution. All other values displayed as zero. Press any key to
 continue.');
                [~,new_dataset_m] =
 display_values(pumpdata_m,rows_v,columns_v);
```

```
                disp(new_dataset_m);
                pause;
                view_again = input('View another range? Y/N as
character');
            end
        elseif (choice == 2) %%View and modify values
            % View data as per previous choice
            inp_filename1_str=['meas','\','pumpdata'];
            modify_data = 'Y';
            while (modify_data == 'Y') || (modify_data == 'y') %Modify
data
                load(inp_filename1_str,'pumpdata_m');
                disp(strcat('Data loaded into workspace. Dataset size
is ...',num2str(size(pumpdata_m,1)),'... features by ...',...
                    num2str(size(pumpdata_m,2)),'... samples.'));
                modify_indices_m = input('Please input data rows
and columns to modify as well as new values in a matrix of the form
[row1,col1,newval1;row2,col2,newval2...].');
                disp(modify_indices_m);
                disp('Modifying values then pausing execution. Press
any key to continue.');
                [~,new_dataset_m] =
replace_values(pumpdata_m,modify_indices_m(:,1),modify_indices_m(:,2),modify_indi
                disp('Old dataset:');
                disp(pumpdata_m);
                disp('New dataset with replacements:');
                disp(new_dataset_m);
                pause;
                %Prompt to save changes or discard
                update_data = input('Save changes? Y/N as character');
                if (update_data == 'Y') || (update_data == 'y')
                    pump_data_m=new_dataset_m; %#ok<NASGU>
                    out_filename1_str=['meas' '\','pumpdata'];
                    save(out_filename1_str,'pumpdata_m');
                    input('Changes saved. Press any key to
continue.');
                elseif (update_data == 'N') || (update_data == 'n')
                    input('Changes NOT saved. Press any key to
continue.');
                end
                modify_data = input('Modify another range? Y/N');
            end
        elseif (choice == 3) %Append or reduce dataset
            resize_dataset = 'Y';
            view_again = 'Y';
            inp_filename1_str=['meas','\','pumpdata'];
            load(inp_filename1_str,'pumpdata_m');
            while (view_again == 'Y') || (view_again == 'y')
                disp(strcat('Data loaded into workspace. Dataset size
is ...',num2str(size(pumpdata_m,1)),'... features by ...',...
                    num2str(size(pumpdata_m,2)),'... samples.'));
                rows_v = input('Please input row numbers to view as a
vector[a,b].');
```

```
                columns_v = input('Please input column numbers to view
as a vector[a,b].');
                disp('Displaying specified range and pausing
execution. All other values displayed as zero. Press any key to
continue.');
                [~,new_dataset_m] =
display_values(pumpdata_m,rows_v,columns_v);
                disp(new_dataset_m);
                pause;
                view_again = input('View another range? Y/N as
character');
            end
            disp('Proceeding to change size of dataset ...');
            resized_dataset_m = pumpdata_m;
            while (resize_dataset == 'Y') || (resize_dataset == 'y')
                num_rows = size(resized_dataset_m,1);
                num_columns = size(resized_dataset_m,2);
                disp(strcat('Data loaded into workspace. Dataset size
is ...',num2str(num_rows),'... features by ...',...
                    num2str(num_columns),'... samples.'));
                action = menu('What do you want to do?','1 = Append
dataset','2 = Extract subset of dataset');
                if action == 1
                    disp('Appending data by measurement to dataset.');
                    append_prompt_str = strcat('Please enter a matrix
of ...',num2str(num_rows),'... rows and desired number of columns in
the form [a,b,c...]');
                    append_measurements_m = input(append_prompt_str);
                    resized_dataset_results_m =
[resized_dataset_m,append_measurements_m];
                    new_rows = size(resized_dataset_results_m,1);
                    new_columns = size(resized_dataset_results_m,2);
                elseif action == 2
                    disp('Extracting subset of dataset.');
                    extract_rows_v = input('Please enter rows of
dataset to extract as a vector of the form [a,b,c...].');
                    extract_rows_v = sort(extract_rows_v);
                    extract_columns_v = input('Please enter columns of
dataset to extract as a vector of the form [a,b,c...].');
                    extract_columns_v = sort(extract_columns_v);
                    resized_dataset_results_m =
resized_dataset_m(extract_rows_v,extract_columns_v);
                    new_rows = size(resized_dataset_results_m,1);
                    new_columns = size(resized_dataset_results_m,2);
                end
                disp(strcat('The resulting dataset
is a matrix with ...',num2str(new_rows),'... rows
and ...',num2str(new_columns),':'));
                disp(resized_dataset_results_m);
                disp('Press any key to continue.');
                pause;
                resize_dataset = input('Resize_dataset further? Y/N as
character');
            end
```

```
            save_resize = input('Save changes and overwrite dataset?
Y/N as character');
            if (save_resize == 'Y') || (save_resize == 'y')
            pumpdata_m = resized_dataset_results_m;
            out_filename2_str=['meas','\','pumpdata'];
            save(out_filename2_str,'pumpdata_m');
            input('Resized data saved. Press any key to continue.');
            elseif (save_resize == 'N') || (save_resize == 'n')
               input('Changes not saved. Press any key to continue.');
            end
        elseif (choice == 4)
            inp_filename3_str=['meas','\','pumpdata_default'];
            load(inp_filename3_str,'pumpdata_default_m');
            pumpdata_m = pumpdata_default_m;
            out_filename2_str=['meas','\','pumpdata'];
            save(out_filename2_str,'pumpdata_m');
            input('Default data restored. Press any key to
continue.');
        elseif (choice == 5) %View only
            view_again = 'Y';
            inp_filename1_str=['meas','\','testdata'];
            load(inp_filename1_str,'testdata_m');
            while (view_again == 'Y') || (view_again == 'y')
                disp(strcat('Data loaded into workspace. Dataset size
is ...',num2str(size(testdata_m,1)),'... features by ...',...
                    num2str(size(testdata_m,2)),'... samples.'));
                rows_v = input('Please input row numbers to view as a
vector[a,b].');
                columns_v = input('Please input column numbers to view
as a vector[a,b].');
                disp('Displaying specified range and pausing
execution. All other values displayed as zero. Press any key to
continue.');
                [~,new_dataset_m] =
display_values(testdata_m,rows_v,columns_v);
                disp(new_dataset_m);
                pause;
                view_again = input('View another range? Y/N as
character');
            end
        elseif (choice == 6) %%View and modify
            % View data as per previous choice
            inp_filename1_str=['meas','\','testdata'];
            modify_data = 'Y';
            while (modify_data == 'Y') || (modify_data == 'y') %Modify
data
                load(inp_filename1_str,'testdata_m');
                disp(strcat('Data loaded into workspace. Dataset size
is ...',num2str(size(testdata_m,1)),'... features by ...',...
                    num2str(size(testdata_m,2)),'... samples.'));
                modify_indices_m = input('Please input data rows
and columns to modify as well as new values in a matrix of the form
[row1,col1,newval1;row2,col2,newval2...].');
                disp(modify_indices_m);
```

```
                disp('Modifying values then pausing execution. Press
any key to continue.');
                [~,new_dataset_m] =
replace_values(testdata_m,modify_indices_m(:,1),modify_indices_m(:,2),modify_indi
                disp('Old dataset:');
                disp(testdata_m);
                disp('New dataset with replacements:');
                disp(new_dataset_m);
                pause;
                %Prompt to save changes or discard
                update_data = input('Save changes? Y/N as character');
                if (update_data == 'Y') || (update_data == 'y')
                    test_data_m=new_dataset_m; %#ok<NASGU>
                    out_filename1_str=['meas' '\','testdata'];
                    save(out_filename1_str,'testdata_m');
                    input('Changes saved. Press any key to
continue.');
                elseif (update_data == 'N') || (update_data == 'n')
                    input('Changes NOT saved. Press any key to
continue.');
                end
                modify_data = input('Modify another range? Y/N as
character');
            end
        elseif (choice == 7) %Append or reduce dataset
            resize_dataset = 'Y';
            view_again = 'Y';
            inp_filename1_str=['meas','\','testdata'];
            load(inp_filename1_str,'testdata_m');
            while (view_again == 'Y') || (view_again == 'y')
                disp(strcat('Data loaded into workspace. Dataset size
is ...',num2str(size(testdata_m,1)),'... features by ...',...
                    num2str(size(testdata_m,2)),'... samples.'));
                rows_v = input('Please input row numbers to view as a
vector[a,b].');
                columns_v = input('Please input column numbers to view
as a vector[a,b].');
                disp('Displaying specified range and pausing
execution. All other values displayed as zero. Press any key to
continue.');
                [~,new_dataset_m] =
display_values(testdata_m,rows_v,columns_v);
                disp(new_dataset_m);
                pause;
                view_again = input('View another range? Y/N as
character');
            end
            disp('Proceeding to change size of dataset ...');
            resized_dataset_m = testdata_m;
            while (resize_dataset == 'Y') || (resize_dataset == 'y')
                num_rows = size(resized_dataset_m,1);
                num_columns = size(resized_dataset_m,2);
                disp(strcat('Data loaded into workspace. Dataset size
is ...',num2str(num_rows),'... features by ...',...
```

```
                    num2str(num_columns),'... samples.'));
                action = menu('What do you want to do?','1 = Append
dataset','2 = Extract subset of dataset');
                if action == 1
                    disp('Appending data by measurement to dataset.');
                    append_prompt_str = strcat('Please enter a matrix
of ...',num2str(num_rows),'... rows and desired number of columns in
the form [a,b,c...]');
                    append_measurements_m = input(append_prompt_str);
                    resized_dataset_results_m =
[resized_dataset_m,append_measurements_m];
                    new_rows = size(resized_dataset_results_m,1);
                    new_columns = size(resized_dataset_results_m,2);
                elseif action == 2
                    disp('Extracting subset of dataset.');
                    extract_rows_v = input('Please enter rows of
dataset to extract as a vector of the form [a,b,c...].');
                    extract_rows_v = sort(extract_rows_v);
                    extract_columns_v = input('Please enter columns of
dataset to extract as a vector of the form [a,b,c...].');
                    extract_columns_v = sort(extract_columns_v);
                    resized_dataset_results_m =
resized_dataset_m(extract_rows_v,extract_columns_v);
                    new_rows = size(resized_dataset_results_m,1);
                    new_columns = size(resized_dataset_results_m,2);
                end
                disp(strcat('The resulting dataset
is a matrix with ...',num2str(new_rows),'... rows
and ...',num2str(new_columns),':'));
                disp(resized_dataset_results_m);
                disp('Press any key to continue.');
                pause;
                resize_dataset = input('Resize_dataset further? Y/N as
character');
            end
            save_resize = input('Save changes and overwrite dataset?
Y/N as character');
            if (save_resize == 'Y') || (save_resize == 'y')
            testdata_m = resized_dataset_results_m;
            out_filename2_str=['meas','\','testdata'];
            save(out_filename2_str,'testdata_m');
            input('Resized data saved. Press any key to continue.');
            elseif (save_resize == 'N') || (save_resize == 'n')
               input('Changes not saved. Press any key to continue.');
            end
         elseif (choice == 8)
            inp_filename3_str=['meas','\','testdata_default'];
            load(inp_filename3_str,'testdata_default_m');
            testdata_m = testdata_default_m;
            out_filename2_str=['meas','\','testdata'];
            save(out_filename2_str,'testdata_m');
            input('Default data restored. Press any key to
continue.');
        end
```

```
    end
    input('Returning to Main Menu. Press any key to continue.');
end
```

Published with MATLAB® R2018b

```
function [multi_attribute_utilities_m] =
 multi_attribute_utilities_atsea_pump(Pos_v)
% multi_attribute_utilities_atsea_pump evaluates the 5 sets of values
 when the vessel is at sea
% In this case, the vessel is at sea
% This function uses survey/pump data
% Important constants are defined (assumptions) and used to evaluate
 MAU. In
% some cases we set values to zero (eg. in the case of pump running
 fine,
% otherwise utility function evaluation doesn't make sense)
%
% Input:       Pos_v                          An 1x8 vector of
 probabilities
% Output:      multi_attribute_utilities_m    A 5x8 matrix of multi
%                                             attribute utility
 values considering
%                                             the vessel is at sea

% CALCULATIONS:
    %Load constants for calculation
    inp_filename_str = ['utils' '\' 'constants_pump'];

 load(inp_filename_str,'constants_m','cost_pump','cost_ship','eng_rate',...

  'insp_time','min_time','num_injured','sched_time','stop_time');
    % Define the attribute value matrices for each lottery, using
 constants_m
    % Lottery 1, Stop pump: 8 x 6 matrix
    lottery1_attr_vals_m = zeros(8,6);
    % Calculate attribute 1, maintenance time
    lottery1_attr_vals_m(1,1) = stop_time + insp_time;
    lottery1_attr_vals_m(2:8,1) = stop_time + sched_time;
    % Calculate attribute 2, risk of pump loss
    lottery1_attr_vals_m(:,2) = (Pos_v'.*cost_pump)./10000;
    %Assuming small risk of pump loss remains (1/10 of original risk)
 since we are leaving pump for
    %maintenance at a later date without inspecting
    % Calculate attribute 3, risk of ship loss
    %We assume this is zero because we are stopping Number 2 pump and
 using
    %Number 1 pump
    % Calculate attribute 4, risk of injury to one person
    %We assume this is zero because we are stopping Number 2 pump and
 using
    %Number 1 pump
    % Calculate attribute 5, overall maintenance cost, just the
 equivalent cost
    % of time spent stopping and inspecting
    lottery1_attr_vals_m(1,5) =
 (lottery1_attr_vals_m(1,1)*eng_rate)/1000;
```

```
    lottery1_attr_vals_m(2,5) =
 (lottery1_attr_vals_m(2,1)*eng_rate)/1000;
    lottery1_attr_vals_m(3,5) =
 (lottery1_attr_vals_m(3,1)*eng_rate)/1000;
    lottery1_attr_vals_m(4,5) =
 (lottery1_attr_vals_m(4,1)*eng_rate)/1000;
    lottery1_attr_vals_m(5,5) =
 (lottery1_attr_vals_m(5,1)*eng_rate)/1000;
    lottery1_attr_vals_m(6,5) =
 (lottery1_attr_vals_m(6,1)*eng_rate)/1000;
    lottery1_attr_vals_m(7,5) =
 (lottery1_attr_vals_m(7,1)*eng_rate)/1000;
    lottery1_attr_vals_m(8,5) =
 (lottery1_attr_vals_m(8,1)*eng_rate)/1000;
    % Calculate attribute 6, Lloyds compliance
    lottery1_attr_vals_m(:,6) = 1; %Compliance is true in all cases
    % Lottery 2, Inspect more than once in 6 hours: 8 x 6 matrix
    lottery2_attr_vals_m = zeros(8,6);
    lottery2_attr_vals_m(:,1) = (5*insp_time) + stop_time +
 sched_time;
    lottery2_attr_vals_m(:,2) = (Pos_v'.*cost_pump)./1000;
    lottery2_attr_vals_m(:,3) = (Pos_v'.*constants_m(:,4))*cost_ship;
    lottery2_attr_vals_m(:,4) =
 (Pos_v'.*constants_m(:,5))*num_injured;
    lottery2_attr_vals_m(:,5) =
 (lottery2_attr_vals_m(:,1).*eng_rate)/1000;
    lottery2_attr_vals_m(:,6) = 1;
    %We assume no risk because of OK condition in row 1
    lottery2_attr_vals_m(1,[2,3,4]) = 0;
    % Lottery 3, Inspect after 6 hours: 8 x 6 matrix
    lottery3_attr_vals_m = zeros(8,6);
    lottery3_attr_vals_m(:,1) = (4*insp_time) + stop_time +
 sched_time;
    lottery3_attr_vals_m(:,2) = ((Pos_v'.*cost_pump)./1000)/
(1.006849315^1);
    lottery3_attr_vals_m(:,3) =
 ((Pos_v'.*constants_m(:,4))*cost_ship)/(1.006849315^1);
    lottery3_attr_vals_m(:,4) =
 (Pos_v'.*constants_m(:,5))*num_injured;
    lottery3_attr_vals_m(:,5) =
 ((lottery3_attr_vals_m(:,1).*eng_rate)/1000)/(1.006849315^1);
    lottery3_attr_vals_m(:,6) = 1;
    %We assume no risk because of OK condition
    lottery3_attr_vals_m(1,[2,3,4]) = 0;
    % Lottery 4, Inspect after 12 hours: 8 x 6 matrix
    lottery4_attr_vals_m = zeros(8,6);
    lottery4_attr_vals_m(:,1) = (2*insp_time) + stop_time +
 sched_time;
    lottery4_attr_vals_m(:,2) = ((Pos_v'.*cost_pump)./1000)/
(1.006849315^2);
    lottery4_attr_vals_m(:,3) =
 ((Pos_v'.*constants_m(:,4))*cost_ship)/(1.006849315^2);
    lottery4_attr_vals_m(:,4) =
 (Pos_v'.*constants_m(:,5))*num_injured;
```

```
    lottery4_attr_vals_m(:,5) =
 ((lottery4_attr_vals_m(:,1).*eng_rate)/1000)/(1.006849315^2);
    lottery4_attr_vals_m(:,6) = 1;
    %We assume no risk because of OK condition
    lottery4_attr_vals_m(1,[2,3,4]) = 0;
    % Lottery 5, Inspect after 24 hours: 8x 6 matrix
    lottery5_attr_vals_m = zeros(8,6);
    lottery5_attr_vals_m(:,1) = insp_time + stop_time + sched_time;
    lottery5_attr_vals_m(:,2) = ((Pos_v'.*cost_pump)./1000)/
(1.006849315^4);
    lottery5_attr_vals_m(:,3) =
 ((Pos_v'.*constants_m(:,4))*cost_ship)/(1.006849315^4);
    lottery5_attr_vals_m(:,4) =
 (Pos_v'.*constants_m(:,5))*num_injured;
    lottery5_attr_vals_m(:,5) =
 ((lottery5_attr_vals_m(:,1).*eng_rate)/1000)/(1.006849315^4);
    lottery5_attr_vals_m(:,6) = 1;
    %We assume no risk because of OK condition
    lottery5_attr_vals_m(1,[2,3,4]) = 0;
    % Create the output matrix and perform calculations
    multi_attribute_utilities_m = zeros(5,8);
    for i = 1:length(Pos_v)
        multi_attribute_utilities_m(1,i) =
 evaluate_MAU_pump(lottery1_attr_vals_m(i,:));
        multi_attribute_utilities_m(2,i) =
 evaluate_MAU_pump(lottery2_attr_vals_m(i,:));
        multi_attribute_utilities_m(3,i) =
 evaluate_MAU_pump(lottery3_attr_vals_m(i,:));
        multi_attribute_utilities_m(4,i) =
 evaluate_MAU_pump(lottery4_attr_vals_m(i,:));
        multi_attribute_utilities_m(5,i) =
 evaluate_MAU_pump(lottery5_attr_vals_m(i,:));
    end
end
```

Published with MATLAB® R2018b

```
function [multi_attribute_utilities_m] =
 multi_attribute_utilities_atsea_test(Pos_v)
% multi_attribute_utilities_atsea_test evaluates the 5 sets of values
 when the vessel is at sea
% In this case, the vessel is at sea
% This function uses test data
% Important constants are defined (assumptions) and used to evaluate
 MAU. In
% some cases we set values to zero (eg. in the case of pump running
 fine,
% otherwise utility function evaluation doesn't make sense)
%
% Input:       Pos_v                          An 1x8 vector of
 probabilities
% Output:      multi_attribute_utilities_m    A 5x8 matrix of multi
%                                              attribute utility
 values considering
%                                              the vessel is at sea

% CALCULATIONS:
    %Load constants for calculation
    inp_filename_str = ['utils' '\' 'constants_test'];

 load(inp_filename_str,'constants_m','cost_pump','cost_ship','eng_rate',...

  'insp_time','min_time','num_injured','sched_time','stop_time');
    % Define the attribute value matrices for each lottery, using
 constants_m
    % Lottery 1, Stop pump: 8 x 6 matrix
    lottery1_attr_vals_m = zeros(8,6);
    % Calculate attribute 1, maintenance time
    lottery1_attr_vals_m(1,1) = stop_time + insp_time;
    lottery1_attr_vals_m(2:8,1) = stop_time + sched_time;
    % Calculate attribute 2, risk of pump loss
    lottery1_attr_vals_m(:,2) = (Pos_v'.*cost_pump)./10000;
    %Assuming small risk of pump loss remains (1/10 of original risk)
 since we are leaving pump for
    %maintenance at a later date without inspecting
    % Calculate attribute 3, risk of ship loss
    %We assume this is zero because we are stopping Number 2 pump and
 using
    %Number 1 pump
    % Calculate attribute 4, risk of injury to one person
    %We assume this is zero because we are stopping Number 2 pump and
 using
    %Number 1 pump
    % Calculate attribute 5, overall maintenance cost, just the
 equivalent cost
    % of time spent stopping and inspecting
    lottery1_attr_vals_m(1,5) =
 (lottery1_attr_vals_m(1,1)*eng_rate)/1000;
```

```
    lottery1_attr_vals_m(2,5) =
 (lottery1_attr_vals_m(2,1)*eng_rate)/1000;
    lottery1_attr_vals_m(3,5) =
 (lottery1_attr_vals_m(3,1)*eng_rate)/1000;
    lottery1_attr_vals_m(4,5) =
 (lottery1_attr_vals_m(4,1)*eng_rate)/1000;
    lottery1_attr_vals_m(5,5) =
 (lottery1_attr_vals_m(5,1)*eng_rate)/1000;
    lottery1_attr_vals_m(6,5) =
 (lottery1_attr_vals_m(6,1)*eng_rate)/1000;
    lottery1_attr_vals_m(7,5) =
 (lottery1_attr_vals_m(7,1)*eng_rate)/1000;
    lottery1_attr_vals_m(8,5) =
 (lottery1_attr_vals_m(8,1)*eng_rate)/1000;
    % Calculate attribute 6, Lloyds compliance
    lottery1_attr_vals_m(:,6) = 1; %Compliance is true in all cases
    % Lottery 2, Inspect more than once in 6 hours: 8 x 6 matrix
    lottery2_attr_vals_m = zeros(8,6);
    lottery2_attr_vals_m(:,1) = (5*insp_time) + stop_time +
 sched_time;
    lottery2_attr_vals_m(:,2) = (Pos_v'.*cost_pump)./1000;
    lottery2_attr_vals_m(:,3) = (Pos_v'.*constants_m(:,4))*cost_ship;
    lottery2_attr_vals_m(:,4) =
 (Pos_v'.*constants_m(:,5))*num_injured;
    lottery2_attr_vals_m(:,5) =
 (lottery2_attr_vals_m(:,1).*eng_rate)/1000;
    lottery2_attr_vals_m(:,6) = 1;
    %We assume no risk because of OK condition in row 1
    lottery2_attr_vals_m(1,[2,3,4]) = 0;
    % Lottery 3, Inspect after 6 hours: 8 x 6 matrix
    lottery3_attr_vals_m = zeros(8,6);
    lottery3_attr_vals_m(:,1) = (4*insp_time) + stop_time +
 sched_time;
    lottery3_attr_vals_m(:,2) = ((Pos_v'.*cost_pump)./1000)/
(1.006849315^1);
    lottery3_attr_vals_m(:,3) =
 ((Pos_v'.*constants_m(:,4))*cost_ship)/(1.006849315^1);
    lottery3_attr_vals_m(:,4) =
 (Pos_v'.*constants_m(:,5))*num_injured;
    lottery3_attr_vals_m(:,5) =
 ((lottery3_attr_vals_m(:,1).*eng_rate)/1000)/(1.006849315^1);
    lottery3_attr_vals_m(:,6) = 1;
    %We assume no risk because of OK condition
    lottery3_attr_vals_m(1,[2,3,4]) = 0;
    % Lottery 4, Inspect after 12 hours: 8 x 6 matrix
    lottery4_attr_vals_m = zeros(8,6);
    lottery4_attr_vals_m(:,1) = (2*insp_time) + stop_time +
 sched_time;
    lottery4_attr_vals_m(:,2) = ((Pos_v'.*cost_pump)./1000)/
(1.006849315^2);
    lottery4_attr_vals_m(:,3) =
 ((Pos_v'.*constants_m(:,4))*cost_ship)/(1.006849315^2);
    lottery4_attr_vals_m(:,4) =
 (Pos_v'.*constants_m(:,5))*num_injured;
```

```
    lottery4_attr_vals_m(:,5) =
 ((lottery4_attr_vals_m(:,1).*eng_rate)/1000)/(1.006849315^2);
    lottery4_attr_vals_m(:,6) = 1;
    %We assume no risk because of OK condition
    lottery4_attr_vals_m(1,[2,3,4]) = 0;
    % Lottery 5, Inspect after 24 hours: 8x 6 matrix
    lottery5_attr_vals_m = zeros(8,6);
    lottery5_attr_vals_m(:,1) = insp_time + stop_time + sched_time;
    lottery5_attr_vals_m(:,2) = ((Pos_v'.*cost_pump)./1000)/
(1.006849315^4);
    lottery5_attr_vals_m(:,3) =
 ((Pos_v'.*constants_m(:,4))*cost_ship)/(1.006849315^4);
    lottery5_attr_vals_m(:,4) =
 (Pos_v'.*constants_m(:,5))*num_injured;
    lottery5_attr_vals_m(:,5) =
 ((lottery5_attr_vals_m(:,1).*eng_rate)/1000)/(1.006849315^4);
    lottery5_attr_vals_m(:,6) = 1;
    %We assume no risk because of OK condition
    lottery5_attr_vals_m(1,[2,3,4]) = 0;
    % Create the output matrix and perform calculations
    multi_attribute_utilities_m = zeros(5,8);
        for i = 1:length(Pos_v)
            multi_attribute_utilities_m(1,i) =
 evaluate_MAU_test(lottery1_attr_vals_m(i,:));
            multi_attribute_utilities_m(2,i) =
 evaluate_MAU_test(lottery2_attr_vals_m(i,:));
            multi_attribute_utilities_m(3,i) =
 evaluate_MAU_test(lottery3_attr_vals_m(i,:));
            multi_attribute_utilities_m(4,i) =
 evaluate_MAU_test(lottery4_attr_vals_m(i,:));
            multi_attribute_utilities_m(5,i) =
 evaluate_MAU_test(lottery5_attr_vals_m(i,:));
        end
end
```

Published with MATLAB® R2018b

```
function [multi_attribute_utilities_m] =
 multi_attribute_utilities_atwharf_pump(Pos_v)
% multi_attribute_utilities_atwharf_pump evaluates multi attribute
 utility when the vessel is stationary at the wharf
% This is the same when nothing else is running in the engine room and
 when
% other machinery is running in the engine room
% This function uses survey/pump data
% Important constants are defined (assumptions) and used to evaluate
 MAU. In
% some cases we set values to zero (eg. in the case of pump running
 fine,
% otherwise utility function evaluation doesn't make sense)
%
% Input:  Pos_v                          A 1x8 vector of probabilities
% Output: multi_attribute_utilities_m    A 5x8 matrix of multi-
attribute
%                                        utilities evaluated
 considering the vessel is
%                                        stationary at the wharf

% CALCULATIONS:
    %Load constants for calculation
    inp_filename_str = ['utils' '\' 'constants_pump'];

 load(inp_filename_str,'constants_m','cost_pump','cost_ship','eng_rate',...

  'insp_time','min_time','num_injured','sched_time','stop_time');
    % Define the attribute value matrices for each lottery, using
 constants_m
    %Lottery 1, Stop pump: 16 x 6 matrix
    lottery1_attr_vals_m = zeros(16,6);
    % Calculate attribute 1, maintenance time
    lottery1_attr_vals_m(1,1) = stop_time + stop_time +
 constants_m(1,1) + min_time;
    lottery1_attr_vals_m(2,1) = stop_time + stop_time +
 constants_m(1,1);
    lottery1_attr_vals_m(3,1) = stop_time + stop_time +
 constants_m(2,1) + min_time;
    lottery1_attr_vals_m(4,1) = stop_time + stop_time +
 constants_m(2,1);
    lottery1_attr_vals_m(5,1) = stop_time + stop_time +
 constants_m(3,1) + min_time;
    lottery1_attr_vals_m(6,1) = stop_time + stop_time +
 constants_m(3,1);
    lottery1_attr_vals_m(7,1) = stop_time + stop_time +
 constants_m(4,1) + min_time;
    lottery1_attr_vals_m(8,1) = stop_time + stop_time +
 constants_m(4,1);
    lottery1_attr_vals_m(9,1) = stop_time + stop_time +
 constants_m(5,1) + min_time;
```

```
   lottery1_attr_vals_m(10,1) = stop_time + stop_time +
constants_m(5,1);
   lottery1_attr_vals_m(11,1) = stop_time + stop_time +
constants_m(6,1) + min_time;
   lottery1_attr_vals_m(12,1) = stop_time + stop_time +
constants_m(6,1);
   lottery1_attr_vals_m(13,1) = stop_time + stop_time +
constants_m(7,1) + min_time;
   lottery1_attr_vals_m(14,1) = stop_time + stop_time +
constants_m(7,1);
   lottery1_attr_vals_m(15,1) = stop_time + stop_time +
constants_m(8,1) + min_time;
   lottery1_attr_vals_m(16,1) = stop_time + stop_time +
constants_m(8,1);
   % Calculate attribute 2, risk of pump loss
   %We assume this is zero because we are stopping the pump to do
maintenance
   % Calculate attribute 3, risk of ship loss
   %We assume this is zero because we are stopping the pump to do
maintenance
   % Calculate attribute 4, risk of injury to one person
   %We assume this is zero because we are stopping the pump to do
maintenance
   % Calculate attribute 5, overall maintenance cost
   lottery1_attr_vals_m(1,5) = ((lottery1_attr_vals_m(1,1)*eng_rate)
+ sum(constants_m(1,2:3)))/1000;
   lottery1_attr_vals_m(2,5) = ((lottery1_attr_vals_m(2,1)*eng_rate)
+ sum(constants_m(1,2:3)))/1000;
   lottery1_attr_vals_m(3,5) = ((lottery1_attr_vals_m(3,1)*eng_rate)
+ sum(constants_m(2,2:3)))/1000;
   lottery1_attr_vals_m(4,5) = ((lottery1_attr_vals_m(4,1)*eng_rate)
+ sum(constants_m(2,2:3)))/1000;
   lottery1_attr_vals_m(5,5) = ((lottery1_attr_vals_m(5,1)*eng_rate)
+ sum(constants_m(3,2:3)))/1000;
   lottery1_attr_vals_m(6,5) = ((lottery1_attr_vals_m(6,1)*eng_rate)
+ sum(constants_m(3,2:3)))/1000;
   lottery1_attr_vals_m(7,5) = ((lottery1_attr_vals_m(7,1)*eng_rate)
+ sum(constants_m(4,2:3)))/1000;
   lottery1_attr_vals_m(8,5) = ((lottery1_attr_vals_m(8,1)*eng_rate)
+ sum(constants_m(4,2:3)))/1000;
   lottery1_attr_vals_m(9,5) = ((lottery1_attr_vals_m(9,1)*eng_rate)
+ sum(constants_m(5,2:3)))/1000;
   lottery1_attr_vals_m(10,5) =
((lottery1_attr_vals_m(10,1)*eng_rate) +
sum(constants_m(5,2:3)))/1000;
   lottery1_attr_vals_m(11,5) =
((lottery1_attr_vals_m(11,1)*eng_rate) +
sum(constants_m(6,2:3)))/1000;
   lottery1_attr_vals_m(12,5) =
((lottery1_attr_vals_m(12,1)*eng_rate) +
sum(constants_m(6,2:3)))/1000;
   lottery1_attr_vals_m(13,5) =
((lottery1_attr_vals_m(13,1)*eng_rate) +
sum(constants_m(7,2:3)))/1000;
```

```
    lottery1_attr_vals_m(14,5) =
 ((lottery1_attr_vals_m(14,1)*eng_rate) +
 sum(constants_m(7,2:3)))/1000;
    lottery1_attr_vals_m(15,5) =
 ((lottery1_attr_vals_m(15,1)*eng_rate) +
 sum(constants_m(8,2:3)))/1000;
    lottery1_attr_vals_m(16,5) =
 ((lottery1_attr_vals_m(16,1)*eng_rate) +
 sum(constants_m(8,2:3)))/1000;
    % Calculate attribute 6, Lloyds compliance
    lottery1_attr_vals_m(:,6) = 1; %Compliance is true in all cases
    % Lottery 2, Inspect more than once in 6 hours: 8 x 6 matrix
    lottery2_attr_vals_m = zeros(8,6);
    lottery2_attr_vals_m(:,1) = (5*insp_time) + stop_time +
 sched_time;
    lottery2_attr_vals_m(:,2) = (Pos_v'.*cost_pump)./1000;
    lottery2_attr_vals_m(:,3) = (Pos_v'.*constants_m(:,4))*cost_ship;
    lottery2_attr_vals_m(:,4) =
 (Pos_v'.*constants_m(:,5))*num_injured;
    lottery2_attr_vals_m(:,5) =
 (lottery2_attr_vals_m(:,1).*eng_rate)/1000;
    lottery2_attr_vals_m(:,6) = 1;
    %We assume no risk because of OK condition in row 1
    lottery2_attr_vals_m(1,[2,3,4]) = 0;
    % Lottery 3, Inspect after 6 hours: 8 x 6 matrix
    lottery3_attr_vals_m = zeros(8,6);
    lottery3_attr_vals_m(:,1) = (4*insp_time) + stop_time +
 sched_time;
    lottery3_attr_vals_m(:,2) = ((Pos_v'.*cost_pump)./1000)/
(1.006849315^1);
    lottery3_attr_vals_m(:,3) =
 ((Pos_v'.*constants_m(:,4))*cost_ship)/(1.006849315^1);
    lottery3_attr_vals_m(:,4) =
 (Pos_v'.*constants_m(:,5))*num_injured;
    lottery3_attr_vals_m(:,5) =
 ((lottery3_attr_vals_m(:,1).*eng_rate)/1000)/(1.006849315^1);
    lottery3_attr_vals_m(:,6) = 1;
    %We assume no risk because of OK condition
    lottery3_attr_vals_m(1,[2,3,4]) = 0;
    % Lottery 4, Inspect after 12 hours: 8 x 6 matrix
    lottery4_attr_vals_m = zeros(8,6);
    lottery4_attr_vals_m(:,1) = (2*insp_time) + stop_time +
 sched_time;
    lottery4_attr_vals_m(:,2) = ((Pos_v'.*cost_pump)./1000)/
(1.006849315^2);
    lottery4_attr_vals_m(:,3) =
 ((Pos_v'.*constants_m(:,4))*cost_ship)/(1.006849315^2);
    lottery4_attr_vals_m(:,4) =
 (Pos_v'.*constants_m(:,5))*num_injured;
    lottery4_attr_vals_m(:,5) =
 ((lottery4_attr_vals_m(:,1).*eng_rate)/1000)/(1.006849315^2);
    lottery4_attr_vals_m(:,6) = 1;
    %We assume no risk because of OK condition
    lottery4_attr_vals_m(1,[2,3,4]) = 0;
```

```
    % Lottery 5, Inspect after 24 hours: 8x 6 matrix
    lottery5_attr_vals_m = zeros(8,6);
    lottery5_attr_vals_m(:,1) = insp_time + stop_time + sched_time;
    lottery5_attr_vals_m(:,2) = ((Pos_v'.*cost_pump)./1000)/
(1.006849315^4);
    lottery5_attr_vals_m(:,3) =
 ((Pos_v'.*constants_m(:,4))*cost_ship)/(1.006849315^4);
    lottery5_attr_vals_m(:,4) =
 (Pos_v'.*constants_m(:,5))*num_injured;
    lottery5_attr_vals_m(:,5) =
 ((lottery5_attr_vals_m(:,1).*eng_rate)/1000)/(1.006849315^4);
    lottery5_attr_vals_m(:,6) = 1;
    %We assume no risk because of OK condition
    lottery5_attr_vals_m(1,[2,3,4]) = 0;
    % Create the output matrix and perform calculations
    multi_attribute_utilities_m = zeros(5,16);
        for i = 1:length(Pos_v)
            %Calculate MAU for A1, utilities(1,:)
            multi_attribute_utilities_m(1,(2*i-1)) =
 evaluate_MAU_pump(lottery1_attr_vals_m(((2*i)-1),:));
            multi_attribute_utilities_m(1,(2*i)) =
 evaluate_MAU_pump(lottery1_attr_vals_m((2*i),:));
            %Calculate MAU for lottery2_attr_vals_m to
 lottery5_attr_vals_m
            multi_attribute_utilities_m(2,i) =
 evaluate_MAU_pump(lottery2_attr_vals_m(i,:));
            multi_attribute_utilities_m(3,i) =
 evaluate_MAU_pump(lottery3_attr_vals_m(i,:));
            multi_attribute_utilities_m(4,i) =
 evaluate_MAU_pump(lottery4_attr_vals_m(i,:));
            multi_attribute_utilities_m(5,i) =
 evaluate_MAU_pump(lottery5_attr_vals_m(i,:));
        end
end
```

Published with MATLAB® R2018b

```
function [multi_attribute_utilities_m] =
 multi_attribute_utilities_atwharf_test(Pos_v)
% multi_attribute_utilities_atwharf_test evaluates multi attribute
 utility when the vessel is stationary at the wharf
% This is the same when nothing else is running in the engine room and
 when
% other machinery is running in the engine room
% This function uses test data
% Important constants are defined (assumptions) and used to evaluate
 MAU. In
% some cases we set values to zero (eg. in the case of pump running
 fine,
% otherwise utility function evaluation doesn't make sense)
%
% Input:   Pos_v                          A 1x8 vector of probabilities
% Output:  multi_attribute_utilities_m    A 5x8 matrix of multi-
attribute
%                                         utilities evaluated
 considering the vessel is
%                                         stationary at the wharf

%CALCULATIONS:
    %Load constants for calculation
    inp_filename_str = ['utils' '\' 'constants_test'];

 load(inp_filename_str,'constants_m','cost_pump','cost_ship','eng_rate',...

  'insp_time','min_time','num_injured','sched_time','stop_time');
    % Define the attribute value matrices for each lottery, using
 constants_m
    % Lottery 1, Stop pump: 16 x 6 matrix
    lottery1_attr_vals_m = zeros(16,6);
    % Calculate attribute 1, maintenance time
    lottery1_attr_vals_m(1,1) = stop_time + stop_time +
 constants_m(1,1) + min_time;
    lottery1_attr_vals_m(2,1) = stop_time + stop_time +
 constants_m(1,1);
    lottery1_attr_vals_m(3,1) = stop_time + stop_time +
 constants_m(2,1) + min_time;
    lottery1_attr_vals_m(4,1) = stop_time + stop_time +
 constants_m(2,1);
    lottery1_attr_vals_m(5,1) = stop_time + stop_time +
 constants_m(3,1) + min_time;
    lottery1_attr_vals_m(6,1) = stop_time + stop_time +
 constants_m(3,1);
    lottery1_attr_vals_m(7,1) = stop_time + stop_time +
 constants_m(4,1) + min_time;
    lottery1_attr_vals_m(8,1) = stop_time + stop_time +
 constants_m(4,1);
    lottery1_attr_vals_m(9,1) = stop_time + stop_time +
 constants_m(5,1) + min_time;
```

```
   lottery1_attr_vals_m(10,1) = stop_time + stop_time +
constants_m(5,1);
   lottery1_attr_vals_m(11,1) = stop_time + stop_time +
constants_m(6,1) + min_time;
   lottery1_attr_vals_m(12,1) = stop_time + stop_time +
constants_m(6,1);
   lottery1_attr_vals_m(13,1) = stop_time + stop_time +
constants_m(7,1) + min_time;
   lottery1_attr_vals_m(14,1) = stop_time + stop_time +
constants_m(7,1);
   lottery1_attr_vals_m(15,1) = stop_time + stop_time +
constants_m(8,1) + min_time;
   lottery1_attr_vals_m(16,1) = stop_time + stop_time +
constants_m(8,1);
   % Calculate attribute 2, risk of pump loss
   %We assume this is zero because we are stopping the pump to do
maintenance
   % Calculate attribute 3, risk of ship loss
   %We assume this is zero because we are stopping the pump to do
maintenance
   % Calculate attribute 4, risk of injury to one person
   %We assume this is zero because we are stopping the pump to do
maintenance
   % Calculate attribute 5, overall maintenance cost
   lottery1_attr_vals_m(1,5) = ((lottery1_attr_vals_m(1,1)*eng_rate)
+ sum(constants_m(1,2:3)))/1000;
   lottery1_attr_vals_m(2,5) = ((lottery1_attr_vals_m(2,1)*eng_rate)
+ sum(constants_m(1,2:3)))/1000;
   lottery1_attr_vals_m(3,5) = ((lottery1_attr_vals_m(3,1)*eng_rate)
+ sum(constants_m(2,2:3)))/1000;
   lottery1_attr_vals_m(4,5) = ((lottery1_attr_vals_m(4,1)*eng_rate)
+ sum(constants_m(2,2:3)))/1000;
   lottery1_attr_vals_m(5,5) = ((lottery1_attr_vals_m(5,1)*eng_rate)
+ sum(constants_m(3,2:3)))/1000;
   lottery1_attr_vals_m(6,5) = ((lottery1_attr_vals_m(6,1)*eng_rate)
+ sum(constants_m(3,2:3)))/1000;
   lottery1_attr_vals_m(7,5) = ((lottery1_attr_vals_m(7,1)*eng_rate)
+ sum(constants_m(4,2:3)))/1000;
   lottery1_attr_vals_m(8,5) = ((lottery1_attr_vals_m(8,1)*eng_rate)
+ sum(constants_m(4,2:3)))/1000;
   lottery1_attr_vals_m(9,5) = ((lottery1_attr_vals_m(9,1)*eng_rate)
+ sum(constants_m(5,2:3)))/1000;
   lottery1_attr_vals_m(10,5) =
((lottery1_attr_vals_m(10,1)*eng_rate) +
sum(constants_m(5,2:3)))/1000;
   lottery1_attr_vals_m(11,5) =
((lottery1_attr_vals_m(11,1)*eng_rate) +
sum(constants_m(6,2:3)))/1000;
   lottery1_attr_vals_m(12,5) =
((lottery1_attr_vals_m(12,1)*eng_rate) +
sum(constants_m(6,2:3)))/1000;
   lottery1_attr_vals_m(13,5) =
((lottery1_attr_vals_m(13,1)*eng_rate) +
sum(constants_m(7,2:3)))/1000;
```

```
    lottery1_attr_vals_m(14,5) =
 ((lottery1_attr_vals_m(14,1)*eng_rate) +
 sum(constants_m(7,2:3)))/1000;
    lottery1_attr_vals_m(15,5) =
 ((lottery1_attr_vals_m(15,1)*eng_rate) +
 sum(constants_m(8,2:3)))/1000;
    lottery1_attr_vals_m(16,5) =
 ((lottery1_attr_vals_m(16,1)*eng_rate) +
 sum(constants_m(8,2:3)))/1000;
    % Calculate attribute 6, Lloyds compliance
    lottery1_attr_vals_m(:,6) = 1; %Compliance is true in all cases
    % Lottery 2, Inspect more than once in 6 hours: 8 x 6 matrix
    lottery2_attr_vals_m = zeros(8,6);
    lottery2_attr_vals_m(:,1) = (5*insp_time) + stop_time +
 sched_time;
    lottery2_attr_vals_m(:,2) = (Pos_v'.*cost_pump)./1000;
    lottery2_attr_vals_m(:,3) = (Pos_v'.*constants_m(:,4))*cost_ship;
    lottery2_attr_vals_m(:,4) =
 (Pos_v'.*constants_m(:,5))*num_injured;
    lottery2_attr_vals_m(:,5) =
 (lottery2_attr_vals_m(:,1).*eng_rate)/1000;
    lottery2_attr_vals_m(:,6) = 1;
    %We assume no risk because of OK condition in row 1
    lottery2_attr_vals_m(1,[2,3,4]) = 0;
    % Lottery 3, Inspect after 6 hours: 8 x 6 matrix
    lottery3_attr_vals_m = zeros(8,6);
    lottery3_attr_vals_m(:,1) = (4*insp_time) + stop_time +
 sched_time;
    lottery3_attr_vals_m(:,2) = ((Pos_v'.*cost_pump)./1000)/
(1.006849315^1);
    lottery3_attr_vals_m(:,3) =
 ((Pos_v'.*constants_m(:,4))*cost_ship)/(1.006849315^1);
    lottery3_attr_vals_m(:,4) =
 (Pos_v'.*constants_m(:,5))*num_injured;
    lottery3_attr_vals_m(:,5) =
 ((lottery3_attr_vals_m(:,1).*eng_rate)/1000)/(1.006849315^1);
    lottery3_attr_vals_m(:,6) = 1;
    %We assume no risk because of OK condition
    lottery3_attr_vals_m(1,[2,3,4]) = 0;
    % Lottery 4, Inspect after 12 hours: 8 x 6 matrix
    lottery4_attr_vals_m = zeros(8,6);
    lottery4_attr_vals_m(:,1) = (2*insp_time) + stop_time +
 sched_time;
    lottery4_attr_vals_m(:,2) = ((Pos_v'.*cost_pump)./1000)/
(1.006849315^2);
    lottery4_attr_vals_m(:,3) =
 ((Pos_v'.*constants_m(:,4))*cost_ship)/(1.006849315^2);
    lottery4_attr_vals_m(:,4) =
 (Pos_v'.*constants_m(:,5))*num_injured;
    lottery4_attr_vals_m(:,5) =
 ((lottery4_attr_vals_m(:,1).*eng_rate)/1000)/(1.006849315^2);
    lottery4_attr_vals_m(:,6) = 1;
    %We assume no risk because of OK condition
    lottery4_attr_vals_m(1,[2,3,4]) = 0;
```

```
    % Lottery 5, Inspect after 24 hours: 8x 6 matrix
    lottery5_attr_vals_m = zeros(8,6);
    lottery5_attr_vals_m(:,1) = insp_time + stop_time + sched_time;
    lottery5_attr_vals_m(:,2) = ((Pos_v'.*cost_pump)./1000)/
(1.006849315^4);
    lottery5_attr_vals_m(:,3) =
 ((Pos_v'.*constants_m(:,4))*cost_ship)/(1.006849315^4);
    lottery5_attr_vals_m(:,4) =
 (Pos_v'.*constants_m(:,5))*num_injured;
    lottery5_attr_vals_m(:,5) =
 ((lottery5_attr_vals_m(:,1).*eng_rate)/1000)/(1.006849315^4);
    lottery5_attr_vals_m(:,6) = 1;
    %We assume no risk because of OK condition
    lottery5_attr_vals_m(1,[2,3,4]) = 0;
    % Create the output matrix and perform calculations
    multi_attribute_utilities_m = zeros(5,16);
        for i = 1:length(Pos_v)
            %Calculate MAU for A1, utilities(1,:)
            multi_attribute_utilities_m(1,(2*i-1)) =
 evaluate_MAU_test(lottery1_attr_vals_m(((2*i)-1),:));
            multi_attribute_utilities_m(1,(2*i)) =
 evaluate_MAU_test(lottery1_attr_vals_m((2*i),:));
            %Calculate MAU for lottery2_attr_vals_m to
 lottery5_attr_vals_m
            multi_attribute_utilities_m(2,i) =
 evaluate_MAU_test(lottery2_attr_vals_m(i,:));
            multi_attribute_utilities_m(3,i) =
 evaluate_MAU_test(lottery3_attr_vals_m(i,:));
            multi_attribute_utilities_m(4,i) =
 evaluate_MAU_test(lottery4_attr_vals_m(i,:));
            multi_attribute_utilities_m(5,i) =
 evaluate_MAU_test(lottery5_attr_vals_m(i,:));
        end
end
```

Published with MATLAB® R2018b

```
function [cond_nw,cond_Pw]=
 objective_space(num_data,Priors_v,max_d_obj)
% objective_space transforms the data into centered normalized
 uncorrelated nonsingular space called objective
%
% Inputs:    num_data   A flag used to select the dataset, 1 2 or 3
 double
%            Priors_v   A vector of prior probabilities
%            max_d_obj  Used to set max number of dimensions in
 objective
%                       space double
%
% Outputs:   cond_nw    A condition number of the resulting matrix
%                       non-weighted priors double
%            cond_Pw    A condition number of the resulting matrix
%                       weighted priors double
% Outputs and other variables are also saved as files

% CALCULATIONS:
    inp_filename_str=['DATA' int2str(num_data) '\parameters'];
    load(inp_filename_str,'mu_c','S_c','Ls_c','WC_c');
    d=size(mu_c{1},1);
    c=length(mu_c);
    n_v=NaN(1,c);
    for k=1:c
        n_v(k)=size(Ls_c{k},2);
    end
    nG=sum(n_v);
    ncum_v=[0 cumsum(n_v)];
    LsG_m=zeros(d,nG);
    WCG_v=zeros(1,nG);
    for k=1:c
        indk_v=(ncum_v(k)+1):ncum_v(k+1);
        LsG_m(:,indk_v)=Ls_c{k};
        WCG_v(indk_v)=WC_c{k};
    end
    mugen_v=zeros(d,1);
    Sgen_m=zeros(d,d);
    for j=1:ncum_v(c+1)
        mugen_v=mugen_v+WCG_v(j)*LsG_m(:,j);
    end
    mugen_v=mugen_v/sum(WCG_v);
    for j=1:ncum_v(c+1)
       Sgen_m=Sgen_m+WCG_v(j)*(LsG_m(:,j)-mugen_v)*((LsG_m(:,j)-
mugen_v))';
    end
    Sgen_m=Sgen_m/(sum(WCG_v)*(nG-1)/nG);
    if nargin<3
        max_d_obj=d;
    end
```

```
 [d_obj,T_m,off_v,lin_err_v,ang_err_v,eigval_v ,rec_eigval_v,acc_eigval_v]=
 sel_positve_eigen(Sgen_m,mugen_v,max_d_obj);
    Table_c=cell(d+1,6);
    Table_c{1,1}='number';
    Table_c{1,2}='eigen value';
    Table_c{1,3}='linear error %';
    Table_c{1,4}='angular  error, deg';
    Table_c{1,5}='recomended eigen value';
    Table_c{1,6}='accepted eigen value';
    for i=1:d
        Table_c{i+1,1}=sprintf('%i',i);
        Table_c{i+1,2}=sprintf('%.3e',eigval_v(i));
        Table_c{i+1,3}=sprintf('%.3e',lin_err_v(i));
        Table_c{i+1,4}=sprintf('%.3e',ang_err_v(i));
        Table_c{i+1,5}=sprintf('%.3e',rec_eigval_v(i));
        Table_c{i+1,6}=sprintf('%.3e',acc_eigval_v(i));
    end
    flag_dash_v=[1;zeros(d-1,1);1];
    flag_dash_v(1+d_obj)=1;
    tabl_m=cell2tabl(Table_c,flag_dash_v);
    disp(tabl_m);
    SumWC_v=zeros(c,1);
    for k=1:c
        SumWC_v(k)=sum(WC_c{k});
    end
    Snw_m=zeros(d,d);
    SPw_m=zeros(d,d);
    for k=1:c
        Snw_m=Snw_m+(n_v(k)-1)*SumWC_v(k)*S_c{k}/n_v(k);
        SPw_m=SPw_m+Priors_v(k)*S_c{k};
    end
    Snw_m=Snw_m*sum(n_v)/(sum(n_v(k))-c)/sum(SumWC_v);
    Snw_obj_m=T_m*Snw_m*T_m';
    cond_nw=cond(Snw_obj_m);
    SPw_obj_m=T_m*SPw_m*T_m';
    cond_Pw=cond(SPw_obj_m);
    out_filename_str=['DATA' int2str(num_data) '\objs'];

 save(out_filename_str,'T_m','off_v','mugen_v','Sgen_m','Priors_v','Snw_m','SPw_m'
end

function
 [d_obj,T_m,off_v,lin_err_v,ang_err_v,eigval_v ,rec_eigval_v,acc_eigval_v]=
 sel_positve_eigen(Sgen_m,mugen_v,max_d_obj)
%
%

    d_orig=size(Sgen_m,1);
    [V_m,Eig_m] = eig(Sgen_m);
    eigval_v=diag(Eig_m);
    [eigval_v,ind_v]=sort(eigval_v,'descend');
    V_m=V_m(:,ind_v);
    % determine d_obj
```

```
    lin_err_v=zeros(d_orig,1);
    ang_err_v=zeros(d_orig,1);
    rec_eigval_v=eigval_v;
    for i=1:d_orig
        if eigval_v(i)>0
            Vt_v=V_m(:,i);
            Vct_v=Sgen_m*V_m(:,i)/eigval_v(i);
            lin_err_v(i)=abs(norm(Vt_v,2)-norm(Vct_v,2))/
norm(Vt_v,2)*100;
            cosae=min(Vt_v'*Vct_v/norm(Vt_v,2)/norm(Vct_v,2),1);
            ang_err_v(i)=acos(cosae)*180/pi;
        else
            lin_err_v(i)=99;
            ang_err_v(i)=180;
        end
    end
    rec_eigval_v(lin_err_v>1 | ang_err_v>1)=0;
    acc_eigval_v=rec_eigval_v;
    acc_eigval_v((max_d_obj+1):d_orig)=0;
    d_obj=sum(acc_eigval_v>0);
    T_m=zeros(d_orig,d_obj);
    for i=1:d_obj
        T_m(:,i)=V_m(:,i)/sqrt(eigval_v(i));
    end
    T_m=T_m';
    off_v=T_m*mugen_v;
end
```

Published with MATLAB® R2018b

```
function [aopt,x0opt,HIopt,exitflag,flagstop,str]=optparam_udec...
    (abeg,x0beg,flagparam,x,u,xd,xu,flagplot,flagfile,flag_lang,DMn)
% OPTPARAM_UDEC optimizes the universal decreasing arctangent utility
 function
%   parameters when the uncertainty interval is on u by minimising WLS
%   goodness-of-fit measure:
%   HI^2(a,x0)=sum{[Um(xi)-uid/2-uiu/2]^2/(uiu-uid)^2}, where
%   Um(xi)=[arctg(a.xu-a.x0)-arctg(a.xi-a.x0)]/[arctg(a.xu-a.x0)-
arctg(a.xd-a.x0)]
%   The optimal utility function parameters are obtained by multi-
dimensional
%   nonlinear minimization of Levenberg-Marquardt. The optimal utility
%   function is plotted, along with its corresponding local risk
 aversion
%   r(x)=-Umopt''(x)/Umopt'(x).
%
%   [aopt,x0opt,HIopt,exitflag,flagstop,str]=optparam_udec...
%
 (abeg,x0beg,flagparam,x,u,xd,xu,flagplot,flagfile,flag_lang,DMn)
%
% abeg          the initial value of the positive parameter a for
 optimization
% x0beg         the initial value of the parameter x0 for optimization
% flagparam     [1 x 2] row-vector indicating which of the utility
%               function parameters (a or x0) to optimize
%               if flagparam=[1 1] optimization on both a and x0 is
%               performed (default if flagparam is empty)
%               if flagparam=[1 0] optimization only on a is performed
%               if flagparam=[0 1] optimization only on x0 is
 performed
%               if flagparam=[0 0] no optimization is performed
% x             [1 x n] row-vector of strictly increasing utility
 quantile
%               values in the open interval between xd and xu, so
 that:
%               xd<x1<x2<...<xn<xu
% u             [2 x n] matrix containing the lower (u(1,:)) and upper
 (u(2,:))
%               boundaries of the utility quantile index uncertainty
 intervals,
%               corresponding to the quantile values in x, where:
%               1>=u(1,1)>=u(1,2)>=...>=u(1,n)>1
%               0>u(2,1)>=u(2,2)>=...>=u(2,n)>=1
%               u(1,j)<u(2,j), for j=1,2,...,n
% xd            the lower boundary of the prize interval, where
 Um(xd)=1
% xu            the upper boundary of the prize interval, where
 Um(xu)=0
% flagplot      flag to plot the resulting utility and local risk
 aversion
%               curves:
```

```
%               if flagplot=1 a figure with both curves is plotted
 (default if
%               flagplot is empty)
%               if flagplot=0, no figure is drawn
% flagfile      flag to save the results from the optimization:
%               if flagfile=1, a data file optpar_u.mat is created to
 store
%               the parameters aopt,x0opt,flagparam, and HIopt
%               if flagfile=0, no data file is created (default if
 flagfile
%               is empty)
% flag_lang     flag for text language:
%               if flag_lang=1, all text prints in Bulgarian
%               if flag_lang=0, all text prints in English (default if
%               flag_lang is empty)
% DMn           indicates the number of the decision maker, whose
%               elicitation results will be plotted (the default value
%               is 0 if DMn is empty)
%
% aopt          the optimized value of the parameter a
% x0opt         the optimized value of the parameter x0
% HIopt         the minimal value of the goodness-of-fit measure
% exitflag      see help for the MATLAB function 'lsqnonlin'
% flagstop      flag for abnormal termination of 'optparam_udec' if
 flagstop~=0
% str           text string containing a message from 'optparam_udec'
 if
%               flagstop~=0
%
%  See also optparam_xdec
%
global h_universal_utility_dec
h_optparam_udec_ic=@optparam_udec_ic;
h_error_udec=@error_udec;
h_lsqnonlin=@lsqnonlin;
h_universal_utility_dec=@universal_utility_dec;
flag_font='Timok';
flag_size=10;
er=0;
if nargin==10
    DMn=0;
elseif nargin==9
    flag_lang=0;
    DMn=0;
elseif nargin==8
    flagfile=0;
    flag_lang=0;
    DMn=0;
elseif nargin==7
    flagplot=1;
    flagfile=0;
    flag_lang=0;
    DMn=0;
elseif nargin<7
```

```
    er=1;
    flagstop=12;
    str='Incorrect number of input parameters (message from
 ''optparam_udec'')';
end
if er==0
    [abeg,x0beg,flagparam,x,u,xd,xu,flagplot,flagfile,flag_lang,DMn...

 ,flagstop,str,er]=h_optparam_udec_ic(abeg,x0beg,flagparam,x,u...
        ,xd,xu,flagplot,flagfile,flag_lang,DMn);
end
if flagstop~=0
    [aopt,x0opt,HIopt,exitflag]=deal([]);
    error(sprintf('%s\ninternal error code er=%i',str,er));
    return
end
param0=[abeg;x0beg];
if sum(flagparam)>0

 opti=optimset('display','off','Jacobian','on','DerivativeCheck','off');
    param=param0(flagparam==1);
    LB=[0;-inf];
    LB=LB(flagparam==1);
    h_error_udec_lsq=@(param)
 error_udec_lsq(param,flagparam,param0,xd,xu,x,u);

 [paramopt,HIopt,dump,exitflag]=h_lsqnonlin(h_error_udec_lsq,param,LB,
[],opti);
    param0opt=param0;
    param0opt(flagparam==1)=paramopt;
    aopt=abs(param0opt(1));
    x0opt=param0opt(2);
else
    HIopt=h_error_udec([],flagparam,param0,xd,xu,x,u);
    aopt=abeg;
    x0opt=x0beg;
    exitflag=0;
end
if flagfile==1
    save('optpar_udec','aopt','x0opt','flagparam','HIopt');
end
if flagplot==1
    if size(x,1)==1
        x=x';
        u=u';
    end
    %clc;
    if flag_lang==0
        txtd={'parameter fixed' 'parameter
 optimized','FontName',flag_font,'FontSize',flag_size};
    elseif flag_lang==1
        txtd={'ôèêñèðàí ïàðàìåòð' 'îïòèìèçèðàí
 ïàðàìåòð','FontName',flag_font,'FontSize',flag_size};
    end
```

```
    disp(sprintf('a=%g (%s)',aopt,txtd{flagparam(1)+1}));
    disp(sprintf('x0=%g (%s)',x0opt,txtd{flagparam(2)+1}));
    disp(sprintf('HIopt=%g ',HIopt));
    figure(1);
    close(1);
    figure(1);
    h=subplot(2,1,1);
    set(h,'FontName',flag_font,'FontSize',flag_size);
    hold on;
    xpl=linspace(min([xd;x]),max([xu;x]),101);
    upl=h_universal_utility_dec(xpl,aopt,x0opt,xd,xu);
    h=plot(xpl,upl,'k-',[x x]',u','k-*',[xd xu],[1 0],'kd');
    ax=[1.1*xd-0.1*xu -0.1*xd+1.1*xu -.05 1.05];
    axis(ax);
    ylabel('\itu\rm(\itx\rm)');
    if flag_lang==0
        legend(h(1:2),{'model','data'},'Location','SouthWest');
        if DMn==0
            title('Utility Function');
        else
            title(['Utility Function for Decision Maker No.'
num2str(DMn)]);
        end
    elseif flag_lang==1
        legend(h(1:2),{'ìîäåë','äàííè'},'Location','SouthWest');
        if DMn==0
            title('Ôóíêöèÿ íà ïîëåçíîñò');
        else
            title(['Ôóíêöèÿ íà ïîëåçíîñò ça âçàìàù ðåøåíèå No.'
num2str(DMn)]);
        end
    end
    h=subplot(2,1,2);
    set(h,'FontName',flag_font,'FontSize',flag_size);
    hold on;
    rpl=2*aopt^2*(xpl-x0opt)./(1+aopt^2*(xpl-x0opt).^2);
    h=plot(xpl,rpl,'k-');
    rplmin=min(rpl);
    rplmax=max(rpl);
    if rplmin<rplmax
        ax=[ax(1) ax(2) 1.1*rplmin-0.1*rplmax -0.1*rplmin+1.1*rplmax];
        axis(ax);
    end
    if flag_lang==0
        title('Local risk aversion');
    elseif flag_lang==1
        title('Ëîêàëüíà ðèñêîôîáíîñò');
    end
    ylabel('\itr\rm(\itx\rm)')
    xlabel('\itx');
    set(gcf,'Position',[232   316   454   352]);
end
end
```

```
function [fun,JACfun]=error_udec_lsq(param,flagparam,param0,xd,xu,x,u)
% ERROR_UDEC_LSQ finds the relative aproximation errors in the
 internal knots
%    and their Jacobiant when the uncertainty interval is on u
%
% [fun,JACfun]=error_udec_lsq(param,flagparam,param0,xd,xu,x,u)
%
% param         empty matrix, real number or [1 x 2] row-vector with
%               values of the optimizing parameters a and/or x0
% flagparam     [1 x 2] row-vector indicating which parameter in
%               param should be used
%               if flagparam=[1 1] then a is set to param(1) and x0 is
 set
%               to param(2)
%               if flagparam=[1 0] then a is set to param(1) and x0 is
 set
%               to param0(2)
%               if flagparam=[0 1] then a is set to param0(1) and x0
 is set
%               to param(1)
%               if flagparam=[0 0] then a is set to param0(1) and x0
 is set
%               to param0(2)
% param0        [1 x 2] row-vector containg the initial values of the
%               parameters a and x0
% xd            the lower boundary of the prize interval, where
 Um(xd)=1
% xu            the upper boundary of the prize interval, where
 Um(xu)=0
% x             [1 x n] row-vector of strictly increasing utility
 quantile
%               values in the open interval between xd and xu, so
 that:
%               xd<x1<x2<...<xn<xu
% u             [2 x n] matrix containing the lower (u(1,:)) and upper
 (u(2,:))
%               boundaries of the utility quantile index uncertainty
 intervals,
%               corresponding to the quantile values in x, where:
%               1>=u(1,1)>=u(1,2)>=...>=u(1,n)>1
%               0>u(2,1)>=u(2,2)>=...>=u(2,n)>=1
%               u(1,j)<u(2,j), for j=1,2,...,n
%
% fun           [n x 1] column-vector with i-th entry the relative
%               aproximation error in the i-th internal knot
%               fun(i)=[Um(xi)-u(i,1)/2-u(i,2)/2]/[u(i,2)-u(i,1)],
 where
%               Um(xi)=[arctg(a.xu-a.x0)-arctg(a.xi-a.x0)]/
[arctg(a.xu-a.x0)-arctg(a.xd-a.x0)]
% JACfun        [n x size(length(param))] matrix with the Jacobiant of
 fun,
%               with respect to the parameters in param
%
%      See also error_xdec_lsq
```

```
global h_universal_utility_dec
param0(flagparam==1)=param;
a=param0(1);
x0=param0(2);
if size(x,1)==1
    x=x';
    u=u';
end
[um,flag_over]=h_universal_utility_dec(x,a,x0,xd,xu);
du=u(:,2)-u(:,1);
umes=(u(:,1)+u(:,2))/2;
umden=atan(a*(xu-x0))-atan(a*(xd-x0));
fun=(um-umes)./du;
if nargout>1
    if flagparam(1)==1 && flag_over==0
        dumda=(xu-x0)/(1+a^2*(xu-x0)^2)-(x-x0)./(1+a^2*(x-x0).^2)...
            -um*((xu-x0)/(1+a^2*(xu-x0)^2)-(xd-x0)/(1+a^2*(xd-x0)^2));
        dfunda=dumda./du/umden;
    elseif flagparam(1)==1 && flag_over==1
        dfunda=x*0;
    end
    if flagparam(2)==1 && flag_over==0
        dumdx0=-a/(1+a^2*(xu-x0)^2)+a./(1+a^2*(x-x0).^2)...
            +um*(a/(1+a^2*(xu-x0)^2)-a/(1+a^2*(xd-x0)^2));
        dfundx0=dumdx0./du/umden;
    elseif flagparam(2)==1 && flag_over==1
        dfundx0=x*0;
    end
    if all(flagparam==[0 0])
        JACfun=[];
    elseif flagparam(1)==0
        JACfun=dfundx0;
    elseif flagparam(2)==0
        JACfun=dfunda;
    elseif all(flagparam==[1 1])
        JACfun=[dfunda,dfundx0];
    end
end
end

function [HI,gHI]=error_udec(param,flagparam,param0,xd,xu,x,u)
% ERROR_UDEC finds the HI^2 goodness-of-fit measure of an utility
 function to
%    data knots, when the uncertainty interval is on u
%
% [HI,gHI]=error_udec(param,flagparam,param0,xd,xu,x,u)
%
% param         possible values of the parameters a and x0, other than
%               those in param0
% flagparam     [1 x 2] row-vector indicating the parameters (a or x0)
 to
%               be used
%               if flagparam=[1 1] both a and x0 are used
%               if flagparam=[1 0] only a is used
```

```
%               if flagparam=[0 1] only x0 is used
% param0        [1 x 2] row-vector containing the initial values of
 the
%               parameters a and x0
% xd            the lower boundary of the prize interval, where
 Um(xd)=1
% xu            the upper boundary of the prize interval, where
 Um(xu)=0
% x             [1 x n] row-vector of strictly increasing utility
 quantile
%               values in the open interval between xd and xu, so
 that:
%               xd<x1<x2<...<xn<xu
% u             [2 x n] matrix containing the lower (u(1,:)) and upper
 (u(2,:))
%               boundaries of the utility quantile index uncertainty
 intervals,
%               corresponding to the quantile values in x, where:
%               1>=u(1,1)>=u(1,2)>=...>=u(1,n)>1
%               0>u(2,1)>=u(2,2)>=...>=u(2,n)>=1
%               u(1,j)<u(2,j), for j=1,2,...,n
%
% HI            the value of HI^2 goodness-of-fit measure:
%               HI=sum{[Um(xi)-u(i,2)/2-u(i,1)/2]^2/[u(i,2)-
u(i,1)]^2}, where
%               Um(xi)=[arctg(a.xu-a.x0)-arctg(a.xi-a.x0)]/
[arctg(a.xu-a.x0)-arctg(a.xd-a.x0)]
% gHI           the HI^2 goodness-of-fit measure gradient
%
%      See also error_xdec

[fun,JACfun]=error_udec_lsq(param,flagparam,param0,xd,xu,x,u);
HI=sum(fun.^2);
if nargout>1
    gHI=2*sum(JACfun.*(fun*ones(1,size(JACfun,2))),1)';
end
end

function
 [abeg,x0beg,flagparam,x,u,xd,xu,flagplot,flagfile,flag_lang,DMn...
        ,flagstop,str,er]=optparam_udec_ic...

 (abeg,x0beg,flagparam,x,u,xd,xu,flagplot,flagfile,flag_lang,DMn)
flagstop=0;
str='';
er=0;
er=chcknmat(abeg,1,1);
if er==0 & abeg<=0
    er=1;
end
if er~=0
    flagstop=1;
    str='Incorrect input parameter abeg (message from
 ''optparam_udec'')';
```

```
 [abeg,x0beg,flagparam,x,u,xd,xu,flagplot,flagfile,flag_lang,DMn]=deal([]);
    return
end
er=0;
er=chcknmat(x0beg,1,1);
if er~=0
    flagstop=2;
    str='Incorrect input parameter x0beg (message from
 ''optparam_udec'')';

 [abeg,x0beg,flagparam,x,u,xd,xu,flagplot,flagfile,flag_lang,DMn]=deal([]);
    return
end
er=0;
if isempty(flagparam)
    flagparam=[1 1];
end
er=chcknmat(flagparam,1,2);
if er==0 & all(flagparam(1)~=[0 1])
    er=10;
elseif er==0 & all(flagparam(2)~=[0 1])
    er=11;
end
if er~=0
    flagstop=3;
    str='Incorrect input parameter flagparam (message from
 ''optparam_udec'')';

 [abeg,x0beg,flagparam,x,u,xd,xu,flagplot,flagfile,flag_lang,DMn]=deal([]);
    return
end
er=0;
if ndims(x)~=2
    er=1;
elseif size(x,1)~=1 | size(x,2)==0
    er=2;
end
if er==0
    er=chcknmat(x,1,size(x,2));
end
if er==0 & any(x(1:(end-1))>=x(2:end))
    er=10;
end
if er~=0
    flagstop=4;
    str='Incorrect input parameter x (message from
 ''optparam_udec'')';

 [abeg,x0beg,flagparam,x,u,xd,xu,flagplot,flagfile,flag_lang,DMn]=deal([]);
    return
end
er=0;
if ndims(u)~=2
```

```
    er=1;
elseif size(u,1)~=2
    er=2;
end
if er==0
    er=chcknmat(u,2,size(x,2));
end
if er==0 & any(u(1,1:(end-1))<u(1,2:end))
    er=10;
elseif er==0 & any(u(2,1:(end-1))<u(2,2:end))
    er=11;
elseif er==0 & any(u(1,:)>=u(2,:))
    er=12;
elseif er==0 & u(1,2)>1
    er=13;
elseif er==0 & u(1,end)<0
    er=14;
end
if er~=0
    flagstop=5;
    str='Incorrect input parameter u (message from
 ''optparam_udec'')';

 [abeg,x0beg,flagparam,x,u,xd,xu,flagplot,flagfile,flag_lang,DMn]=deal([]);
    return
end
er=0;
er=chcknmat(xd,1,1);
if er==0 & x(1)<=xd
    er=10;
end
if er~=0
    flagstop=6;
    str='Incorrect input parameter xd (message from
 ''optparam_udec'')';

 [abeg,x0beg,flagparam,x,u,xd,xu,flagplot,flagfile,flag_lang,DMn]=deal([]);
    return
end
er=0;
er=chcknmat(xu,1,1);
if er==0 & xd>=xu
    er=10;
end
if er==0 & x(end)>=xu
    er=11;
end
if er~=0
    flagstop=7;
    str='Incorrect input parameter xu (message from
 ''optparam_udec'')';

 [abeg,x0beg,flagparam,x,u,xd,xu,flagplot,flagfile,flag_lang,DMn]=deal([]);
    return
```

```
end
er=0;
[er,flagplot]=chcknint(flagplot,[0 1],1);
if er~=0
    flagstop=8;
    str='Incorrect input parameter flagplot (message from
 ''optparam_udec'')';

 [abeg,x0beg,flagparam,x,u,xd,xu,flagplot,flagfile,flag_lang,DMn]=deal([]);
    return
end
er=0;
[er,flagfile]=chcknint(flagfile,[0 1],0);
if er~=0
    flagstop=9;
    str='Incorrect input parameter flagfile (message from
 ''optparam_udec'')';

 [abeg,x0beg,flagparam,x,u,xd,xu,flagplot,flagfile,flag_lang,DMn]=deal([]);
    return
end
er=0;
[er,flag_lang]=chcknint(flag_lang,[0 1],0);
if er~=0
    flagstop=10;
    str='Incorrect input parameter flag_lang (message from
 ''optparam_udec'')';

 [abeg,x0beg,flagparam,x,u,xd,xu,flagplot,flagfile,flag_lang,DMn]=deal([]);
    return
end
er=0;
[er,DMn]=chcknint(DMn,0,0);
if er~=0
    flagstop=11;
    str='Incorrect input parameter DMn (message from
 ''optparam_udec'')';

 [abeg,x0beg,flagparam,x,u,xd,xu,flagplot,flagfile,flag_lang,DMn]=deal([]);
    return
end
end
```

Published with MATLAB® R2018b

```
function [aopt,x0opt,HIopt,exitflag,flagstop,str]=optparam_xdec...
    (abeg,x0beg,flagparam,x,u,xd,xu,flagplot,flagfile,flag_lang,DMn)
% OPTPARAM_XDEC optimizes the universal decreasing arctangent utility
%   function parameters when the uncertainty interval is on x by
 minimising
%   WLS goodness-of-fit measure:
%   HI^2(a,x0)=sum{[Xm(ui)-xid/2-xiu/2]^2/(xiu-xid)^2}, where
%   Xm(ui)=tan[(1-ui)*atan(a.xu-a.x0))+ui*atan(a.xd-a.x0)]/a+x0
%   The optimal utility function parameters are obtained by
 multidimensional
%   nonlinear minimization of Levenberg-Marquardt. The optimal utility
%   function is plotted, along with its corresponding local risk
 aversion
%   r(x)=-Umopt''(x)/Umopt'(x), where Umopt(.) is the inverse of
 Xmopt(.)
%
% [aopt,x0opt,HIopt,exitflag,flagstop,str]=optparam_xdec...
%    (abeg,x0beg,flagparam,x,u,xd,xu,flagplot,flagfile,flag_lang,DMn)
%
% abeg          the initial value of the positive parameter a for
 optimization
% x0beg         the initial value of the parameter x0 for optimization
% flagparam     [1 x 2] row-vector indicating which of the utility
%               function parameters (a or x0) to be optimized
%               if flagparam=[1 1] optimization on both a and x is
%               performed (default if flagparam is empty)
%               if flagparam=[1 0] optimization only on a is performed
%               if flagparam=[0 1] optimization only on x0 is
 performed
%               if flagparam=[0 0] no optimization is performed
% x             [2 x n] matrix containing the lower (x(1,:)) and upper
 (x(2,:))
%               boundaries of the utility quantile uncertainty
 intervals,
%               so that:
%               xd<=x(1,1)<=x(1,2)<=...<=x(1,n)<xu
%               xd<x(2,1)<=x(2,2)<=...<=x(2,n)<=xu
%               x(1,j)<x(2,j), for j=1,2,...,n
% u             [1 x n] row-vector of strictly decreasing utility
 quantile
%               index values in the open interval between 0 and 1,
%               corresponding to the quantile index values in x,
 where:
%               1>u1>u2>...>un>0
% xd            the lower boundary of the prize interval, where
 Xm(0)=xu
% xu            the upper boundary of the prize interval, where
 Xm(1)=xd
% flagplot      flag to plot the resulting utility and local risk
 aversion
%               curves:
```

```
%               if flagplot=1 a figure with both curves is drawn
 (default if
%               flagplot is empty)
%               if flagplot=0, no figure is drawn
% flagfile      flag to save the results from the optimization:
%               if flagfile=1, a data file optpar_xdec.mat is created
 to store
%               the parameters aopt,x0opt,flagparam, and HIopt
%               if flagfile=0, no data file is created (default if
 flagfile
%               is empty)
% flag_lang     flag for text language:
%               if flag_lang=1, all text prints in Bulgarian
%               if flag_lang=0, all text prints in English (default if
%               flag_lang is empty)
% DMn           indicates the number of the decision maker, whose
%               elicitation results will be plotted (the default value
%               is 0 if DMn is empty)
%
% aopt          the optimized value of the parameter a
% x0opt         the optimized value of the parameter x0
% HIopt         the minimal of the goodness-of-fit measure
% exitflag      see help for lsqnonlin
% flagstop      flag for abnormal termination of 'optparam_xdec' if
 flagstop~=0
% str           text string containing a message from 'optparam_xdec'
 if
%               flagstop~=0
%
%  See also optparam_udec
%
global h_universal_utility_inv_dec
h_optparam_xdec_ic=@optparam_xdec_ic;
h_error_xdec=@error_xdec;
h_lsqnonlin=@lsqnonlin;
h_universal_utility_inv_dec=@universal_utility_inv_dec;
flag_font='Timok';
flag_size=10;
er=0;
if nargin==10
    DMn=0;
elseif nargin==9
    flag_lang=0;
    DMn=0;
elseif nargin==8
    flagfile=0;
    flag_lang=0;
    DMn=0;
elseif nargin==7
    flagplot=1;
    flagfile=0;
    flag_lang=0;
    DMn=0;
elseif nargin<7
```

```
    er=1;
    flagstop=12;
    str='Incorrect number of input parameters (message from
 ''optparam_xdec'')';
end
if er==0
    [abeg,x0beg,flagparam,x,u,xd,xu,flagplot,flagfile,flag_lang,DMn...

 ,flagstop,str,er]=h_optparam_xdec_ic(abeg,x0beg,flagparam,x,u...
        ,xd,xu,flagplot,flagfile,flag_lang,DMn);
end
if flagstop~=0
    [aopt,x0opt,HIopt,exitflag]=deal([]);
    error(sprintf('%s\ninternal error code er=%i',str,er));
    return
end
param0=[abeg;x0beg];
if sum(flagparam)>0

 opti=optimset('display','off','Jacobian','on','DerivativeCheck','off');
    param=param0(flagparam==1);
    LB=[0;-inf];
    LB=LB(flagparam==1);
    h_error_xdec_lsq=@(param)
 error_xdec_lsq(param,flagparam,param0,xd,xu,x,u);

 [paramopt,HIopt,dump,exitflag]=h_lsqnonlin(h_error_xdec_lsq,param,LB,
[],opti);
    param0opt=param0;
    param0opt(flagparam==1)=paramopt;
    aopt=abs(param0opt(1));
    x0opt=param0opt(2);
else
    HIopt=h_error_xdec([],flagparam,param0,xd,xu,x,u);
    aopt=abeg;
    x0opt=x0beg;
    exitflag=0;
end
if flagfile==1
    save('optpar_xdec','aopt','x0opt','flagparam','HIopt');
end
if flagplot==1
    if size(u,1)==1
        x=x';
        u=u';
    end
    %clc;
    if flag_lang==0
        txtd={'parameter fixed' 'parameter
 optimized','FontName',flag_font,'FontSize',flag_size};
    elseif flag_lang==1
        txtd={'ôèêñèðàí ïàðàìåòð' 'îïòèìèçèðàí
 ïàðàìåòð','FontName',flag_font,'FontSize',flag_size};
    end
```

```
    disp(sprintf('a=%g (%s)',aopt,txtd{flagparam(1)+1}));
    disp(sprintf('x0=%g (%s)',x0opt,txtd{flagparam(2)+1}));
    disp(sprintf('HIopt=%g ',HIopt));
    figure(1);
    close(1);
    figure(1);
    h=subplot(2,1,1);
    set(h,'FontName',flag_font,'FontSize',flag_size);
    hold on;
    upl=linspace(min([0;u]),max([1;u]),101);
    xpl=h_universal_utility_inv_dec(upl,aopt,x0opt,xd,xu);
    h=plot(xpl,upl,'k-',x',[u u]','k-*',[xd xu],[1 0],'kd');
    ax=[1.1*xd-0.1*xu -0.1*xd+1.1*xu -.05 1.05];
    axis(ax);
    ylabel('\itu\rm(\itx\rm)');
    if flag_lang==0
        legend(h(1:2),{'model','data'},'Location','SouthWest');
        if DMn==0
            title('Utility Function');
        else
            title(['Utility Function for Decision Maker No.'
num2str(DMn)]);
        end
    elseif flag_lang==1
        legend(h(1:2),{'ìîäåë','äàííè'},'Location','SouthWest');
        if DMn==0
            title('Ôóíêöèÿ íà ïîëåçíîñò');
        else
            title(['Ôóíêöèÿ íà ïîëåçíîñò ça âçàìàù ðåøåíèå No.'
num2str(DMn)]);
        end
    end
    h=subplot(2,1,2);
    set(h,'FontName',flag_font,'FontSize',flag_size);
    hold on;
    rpl=2*aopt^2*(xpl-x0opt)./(1+aopt^2*(xpl-x0opt).^2);
    h=plot(xpl,rpl,'k-');
    rplmin=min(rpl);
    rplmax=max(rpl);
    if rplmin<rplmax
        ax=[ax(1) ax(2) 1.1*rplmin-0.1*rplmax -0.1*rplmin+1.1*rplmax];
        axis(ax);
    end
    if flag_lang==0
        title('Local risk aversion');
    elseif flag_lang==1
        title('Ëîêàëíà ðèñêîôîáíîñò');
    end
    ylabel('\itr\rm(\itx\rm)');
    xlabel('\itx');
    set(gcf,'Position',[232   316   454   352]);
end
end
```

```
function [fun,JACfun]=error_xdec_lsq(param,flagparam,param0,xd,xu,x,u)
% ERROR_XDEC_LSQ finds the relative aproximation errors  in the
 internal knots
%    and their Jacobiant when the uncertainty interval is on x
%
% [fun,JACfun]=error_xdec_lsq(param,flagparam,param0,xd,xu,x,u)
%
% param         empty matrix, real number or [1 x 2] row-vector with
%               values of the optimizing parameters a and/or x0
% flagparam     [1 x 2] row-vector indicating which parameter in
%               param should be used
%               if flagparam=[1 1] then a is set to param(1) and x0 is
 set
%               to param(2)
%               if flagparam=[1 0] then a is set to param(1) and x0 is
 set
%               to param0(2)
%               if flagparam=[0 1] then a is set to param0(1) and x0
 is set
%               to param(1)
%               if flagparam=[0 0] then a is set to param0(1) and x0
 is set
%               to param0(2)
% param0        [1 x 2] row-vector containg the initial values of the
%               parameters a and x0
% xd            the lower boundary of the prize interval, where
 Xm(0)=xd
% xu            the upper boundary of the prize interval, where
 Xm(1)=xu
% x             [2 x n] matrix containing the lower (x(1,:)) and upper
 (x(2,:))
%               boundaries of the utility quantile uncertainty
 intervals,
%               so that:
%               xd<=x(1,1)<=x(1,2)<=...<=x(1,n)<xu
%               xd<x(2,1)<=x(2,2)<=...<=x(2,n)<=xu
%               x(1,j)<x(2,j), for j=1,2,...,n
% u             [1 x n] row-vector of strictly decreasing utility
 quantile
%               index values in the open interval between 0 and 1,
%               corresponding to the quantile index values in x,
 where:
%               1>u1>u2>...>un>0

%
% fun           [n x 1] column-vector with i-th entry the relative
%               aproximation error in the i-th internal knot
%               fun(i)=[Xm(ui)-x(i,1)/2-x(i,2)/2]/[x(i,2)-x(i,1)],
 where
%               Xm(ui)=tan[ui*atan(a.xu-a.x0))+(1-ui)*atan(a.xd-
a.x0)]/a+x0
% JACfun        [n x size(length(param))] matrix with the Jacobiant of
 fun,
%               with respect to the parameters in param
```

```
%
%      See also error_udec_lsq
global h_universal_utility_inv_dec
param0(flagparam==1)=param;
a=param0(1);
x0=param0(2);
if size(u,1)==1
    x=x';
    u=u';
end
[xm,ang,flag_over]=h_universal_utility_inv_dec(u,a,x0,xd,xu);
dx=x(:,2)-x(:,1);
xmes=(x(:,2)+x(:,1))/2;
fun=(xm-xmes)./dx;
if nargout>1
    if flagparam(1)==1 && flag_over==0
        dxmda=((1-u)*(xu-x0)/(1+a^2*(xu-x0)^2)+u*(xd-x0)/(1+a^2*(xd-
x0)^2))./cos(ang).^2 ...
            -(xm-x0);
        dfunda=dxmda./dx/a;
    elseif flagparam(1)==1 && flag_over==1
        dfunda=u*0;
    end
    if flagparam(2)==1 && flag_over==0
        dxmdx0=(-(1-u)/(1+a^2*(xu-x0)^2)-u/(1+a^2*(xd-x0)^2))./
cos(ang).^2+1;
        dfundx0=dxmdx0./dx;
    elseif flagparam(2)==1 && flag_over==1
        dfundx0=u*0;
    end
    if all(flagparam==[0 0])
        JACfun=[];
    elseif flagparam(1)==0
        JACfun=dfundx0;
    elseif flagparam(2)==0
        JACfun=dfunda;
    elseif all(flagparam==[1 1])
        JACfun=[dfunda,dfundx0];
    end
end
end

function [HI,gHI]=error_xdec(param,flagparam,param0,xd,xu,x,u)
% ERROR_XDEC finds the HI^2 goodness-of-fit measure of an utility
 function to
%    data knots, when the uncertainty interval is on x
%
% HI=error_xdec(param,flagparam,param0,xd,xu,x,u)
%
% param         possible values of the parameters a and x0, other than
%               those in param0
% flagparam     [1 x 2] row-vector indicating the parameters (a or x0)
 to
%               be used
```

```
%               if flagparam=[1 1] both a and x0 are used
%               if flagparam=[1 0] only a is used
%               if flagparam=[0 1] only x0 is used
% param0        [1 x 2] row-vector containing the initial values of
 the
%               parameters a and x0
% xd            the lower boundary of the prize interval, where
 Xm(0)=xd
% xu            the upper boundary of the prize interval, where
 Xm(1)=xu
% x             [2 x n] matrix containing the lower (x(1,:)) and upper
 (x(2,:))
%               boundaries of the utility quantile uncertainty
 intervals,
%               so that:
%               xd<=x(1,1)<=x(1,2)<=...<=x(1,n)<xu
%               xd<x(2,1)<=x(2,2)<=...<=x(2,n)<=xu
%               x(1,j)<x(2,j), for j=1,2,...,n
% u             [1 x n] row-vector of strictly decreasing utility
 quantile
%               index values in the open interval between 0 and 1,
%               corresponding to the quantile index values in x,
 where:
%               1>u1>u2>...>un>0
%
% HI            the value of HI^2 goodness-of-fit measure:
%               HI=sum{[Xm(ui)-x(i,2)/2-x(i,1)/2]^2/[x(i,2)-
x(i,1)]^2}, where
%               Xm(ui)=tan[ui*atan(a.xu-a.x0))+(1-ui)*atan(a.xd-
a.x0)]/a+x0
%
%      See also error_udec
[fun,JACfun]=error_xdec_lsq(param,flagparam,param0,xd,xu,x,u);
HI=sum(fun.^2);
if nargout>1
    gHI=2*sum(JACfun.*(fun*ones(1,size(JACfun,2))),1)';
end
end

function
 [abeg,x0beg,flagparam,x,u,xd,xu,flagplot,flagfile,flag_lang,DMn...
        ,flagstop,str,er]=optparam_xdec_ic...

 (abeg,x0beg,flagparam,x,u,xd,xu,flagplot,flagfile,flag_lang,DMn)
flagstop=0;
str='';
er=0;
er=chcknmat(abeg,1,1);
if er==0 & abeg<=0
    er=1;
end
if er~=0
    flagstop=1;
```

```
    str='Incorrect input parameter abeg (message from
 ''optparam_xdec'')';

 [abeg,x0beg,flagparam,x,u,xd,xu,flagplot,flagfile,flag_lang,DMn]=deal([]);
    return
end
er=0;
er=chcknmat(x0beg,1,1);
if er~=0
    flagstop=2;
    str='Incorrect input parameter x0beg (message from
 ''optparam_xdec'')';

 [abeg,x0beg,flagparam,x,u,xd,xu,flagplot,flagfile,flag_lang,DMn]=deal([]);
    return
end
er=0;
if isempty(flagparam)
    flagparam=[1 1];
end
er=chcknmat(flagparam,1,2);
if er==0 & all(flagparam(1)~=[0 1])
    er=10;
elseif er==0 & all(flagparam(2)~=[0 1])
    er=11;
end
if er~=0
    flagstop=3;
    str='Incorrect input parameter flagparam (message from
 ''optparam_xdec'')';

 [abeg,x0beg,flagparam,x,u,xd,xu,flagplot,flagfile,flag_lang,DMn]=deal([]);
    return
end
er=0;
if ndims(x)~=2
    er=1;
elseif size(x,1)~=2 | size(x,2)==0
    er=2;
end
if er==0
    er=chcknmat(x,2,size(x,2));
end
if er==0 & any(x(1,1:(end-1))>x(1,2:end))
    er=10;
elseif er==0 & any(x(2,1:(end-1))>x(2,2:end))
    er=11;
elseif er==0 & any(x(1,:)>=x(2,:))
    er=12;
end
if er~=0
    flagstop=5;
    str='Incorrect input parameter x (message from
 ''optparam_xdec'')';
```

```
 [abeg,x0beg,flagparam,x,u,xd,xu,flagplot,flagfile,flag_lang,DMn]=deal([]);
    return
end
er=0;
if ndims(u)~=2
    er=1;
elseif size(u,1)~=1
    er=2;
end
if er==0
    er=chcknmat(u,1,size(x,2));
end
if er==0 & any(u(1:(end-1))<=u(2:end))
    er=10;
elseif er==0 & u(1)>=1
    er=11;
elseif er==0 & u(end)<=0
    er=12;
end
if er~=0
    flagstop=4;
    str='Incorrect input parameter u (message from
 ''optparam_xdec'')';

 [abeg,x0beg,flagparam,x,u,xd,xu,flagplot,flagfile,flag_lang,DMn]=deal([]);
    return
end
er=0;
er=chcknmat(xd,1,1);
if er==0 & x(1,1)<xd
    er=10;
end
if er~=0
    flagstop=6;
    str='Incorrect input parameter xd (message from
 ''optparam_xdec'')';

 [abeg,x0beg,flagparam,x,u,xd,xu,flagplot,flagfile,flag_lang,DMn]=deal([]);
    return
end
er=0;
er=chcknmat(xu,1,1);
if er==0 & xd>=xu
    er=10;
end
if er==0 & x(2,end)>xu
    er=11;
end
if er~=0
    flagstop=7;
    str='Incorrect input parameter xu (message from
 ''optparam_xdec'')';
```

```
 [abeg,x0beg,flagparam,x,u,xd,xu,flagplot,flagfile,flag_lang,DMn]=deal([]);
    return
end
er=0;
[er,flagplot]=chcknint(flagplot,[0 1],1);
if er~=0
    flagstop=8;
    str='Incorrect input parameter flagplot (message from
 ''optparam_xdec'')';

 [abeg,x0beg,flagparam,x,u,xd,xu,flagplot,flagfile,flag_lang,DMn]=deal([]);
    return
end
er=0;
[er,flagfile]=chcknint(flagfile,[0 1],0);
if er~=0
    flagstop=9;
    str='Incorrect input parameter flagfile (message from
 ''optparam_xdec'')';

 [abeg,x0beg,flagparam,x,u,xd,xu,flagplot,flagfile,flag_lang,DMn]=deal([]);
    return
end
er=0;
[er,flag_lang]=chcknint(flag_lang,[0 1],0);
if er~=0
    flagstop=10;
    str='Incorrect input parameter flag_lang (message from
 ''optparam_xdec'')';

 [abeg,x0beg,flagparam,x,u,xd,xu,flagplot,flagfile,flag_lang,DMn]=deal([]);
    return
end
er=0;
[er,DMn]=chcknint(DMn,0,0);
if er~=0
    flagstop=11;
    str='Incorrect input parameter DMn (message from
 ''optparam_xdec'')';

 [abeg,x0beg,flagparam,x,u,xd,xu,flagplot,flagfile,flag_lang,DMn]=deal([]);
    return
end
end
```

Published with MATLAB® R2018b

```
function [S_c,mu_c] = Param_estim(num_data)
% Param_estima calculates the mean vectors and covarance matrices of
 the classes
% Input:    num_data  A flag indicating which data to use, either 1,
 2, 3
%                     or 4
% Outputs:  S_c       A cell array containing covariance matrices for
 all classes
%           mu_c      A cell array containing the mean vectors for all
 classes
% Outputs are also saved to a file

% CALCULATIONS:
    inp_filename_str=['DATA' int2str(num_data) '\learning_data'];

 load(inp_filename_str,'Ls_1_m','Ls_2_m','Ls_3_m','Ls_4_m','Ls_5_m',...
        'Ls_6_m','Ls_7_m','Ls_8_m');
    load(inp_filename_str,'WC1_v','WC2_v','WC3_v','WC4_v','WC5_v',...
        'WC6_v','WC7_v','WC8_v');
    load(inp_filename_str,'class_names_c','features_c');
    Ls_c={Ls_1_m,Ls_2_m,Ls_3_m,Ls_4_m,Ls_5_m,Ls_6_m,Ls_7_m,Ls_8_m}';
    WC_c={WC1_v,WC2_v,WC3_v,WC4_v,WC5_v,WC6_v,WC7_v,WC8_v}';
    c=8;
    d=size(features_c,2);
    mu_c=cell(c,1);
    S_c=cell(c,1);
    for k=1:c
        muk_v=zeros(d,1);
        Sk_m=zeros(d,d);
        Lsk_m=Ls_c{k};
        WCk_v=WC_c{k};
        nk=size(Lsk_m,2);
        for j=1:nk
            muk_v=muk_v+WCk_v(j)*Lsk_m(:,j);
        end
        muk_v=muk_v/sum(WCk_v);
        for j=1:nk
            Sk_m=Sk_m+WCk_v(j)*(Lsk_m(:,j)-muk_v)*((Lsk_m(:,j)-
muk_v))';
        end
        Sk_m=Sk_m/(sum(WCk_v)*(nk-1)/nk);
        mu_c{k}=muk_v;
        S_c{k}=Sk_m;
    end
    out_filename_str=['DATA' int2str(num_data) '\parameters'];
    save(out_filename_str,'mu_c','S_c','Ls_c','WC_c');
end
```

Published with MATLAB® R2018b

```
function [confusion_m,certainty_m,doubt_m] =
 performance_est(result_sample_c)
% performance_est analyses data according to the number of features
 specified
% It produces confusionMat, certaintyMat and doubtMat
%
% Input: result_sample_c   A cell array of posterior
%                          probabilities of all samples in each class
%                          Length of array is number of classes
%
% Outputs: confusion_m      A c x c matrix representing
%                          confusion of classifier used to predict the
 class
%                          numbers of s samples
%          certainty_m     A c x c representing
%                          percentage certainty of classifier used to
 predict
%                          the class numbers of s samples
%          doubt_m         A c x c matrix representing
%                          percentage doubt of classifier used to
 predict the
%                          class numbers of s samples

%CALCULATIONS:
    %Define number of classes
    c = length(result_sample_c);
    %Create the results matrices
    confusion_m = zeros(c,c);
    certainty_m = confusion_m;
    doubt_m = confusion_m;
    %Perform calculations
    for k = 1:c
        nk = size(result_sample_c{k},2);
        current_class_data=zeros(c+1,nk);
        current_class_data(1:c,:) = result_sample_c{k};
        [~,classified_labels]=max(current_class_data,[],1);
        current_class_data(c+1,:) = classified_labels;
        for j = 1:c
            %1. Populate confusion matrix
            confusion_m(k,j) = ((sum(current_class_data(end,:) == j))/
nk)*100;
            %2. Populate certainty matrix
            certainty_data_subset = current_class_data(:,
(find(current_class_data(end,:) == j)));
            if isempty(certainty_data_subset) == 0
                certainty_m(k,j) =
 (mean(certainty_data_subset(j,:)))*100;
            elseif isempty(certainty_data_subset) == 1
                certainty_m(k,j) = NaN;
            end
            %3. Populate doubt matrix
            if k == j
```

```
                if j == 1
                    doubt_data_subset = certainty_data_subset((k
+1:end-1),:);
                elseif j > 1
                    doubt_data_subset = certainty_data_subset([1:k-1 k
+1:(end-1)],:);
                end
            elseif k ~= j
                doubt_data_subset = certainty_data_subset(k,:);
            end
            if isempty(doubt_data_subset) == 0
                doubt_m(k,j) = (mean(max(doubt_data_subset,
[],1)))*100;
            elseif isempty(doubt_data_subset) == 1
                doubt_m(k,j) = NaN;
            end
        end
    end
end
```

Published with MATLAB® R2018b

```
function [ probabilities_m ] = probability_matrix(num_rows)
% probability_matrix creates an num_rows x 8 matrix of probabilites
% Numbers are randomly generated and placed within the matrix before
% each row is normalised to produce probabilites

% Input:    num_rows        An integer specifying how many rows n of
 the nx8
%                           probability matrix to generate
% Output:   probabilities_m A num_rows x 8 matrix of randomly
 generated
%                           probability values

% CALCULATIONS:
    probabilities_m = zeros(num_rows,8);
    random_nums_m=abs(randn(num_rows,8));
    for i = 1:num_rows
        probabilities_m(i,:) = random_nums_m(i,:)/
sum(random_nums_m(i,:));
    end
end
```

Published with MATLAB® R2018b

```
function
 [exp_util]=recursive_ICOL(Pos_v,extra_prob_v,multi_attribute_utilities_v)
% recursive_ICOL calculates the expected utility of a recursive
 infinite compound lottery
% Inputs must have same length
% A certain structure of ICOL is required, refer to pump maintenance
% paper/documents and 'Expected Utility Analysis of Infinite Compound
 Lotteries'
% 2018)
%
% Inputs: Pos_v                       1x8 vector of state posterior
 probabilities
%         extra_prob_v                1x8 vector representing
 additional
%                                     probabilities from the second
 level of
%                                     the original ICOL eg. P(pump
 stopping |
%                                     state)
%         multi_attribute_utilities_v 1x8 vector of multi attribute
 utilities
%
% Output: exp_util                    Expected utility of the lottery
%                                     double

% CALCULATIONS:
    %Calculate the aggregate probabilities in the ficticious simple OL
    aggregate_prob_v = Pos_v.*extra_prob_v;
    norm_aggregate_prob_v = aggregate_prob_v/sum(aggregate_prob_v);
    %In this case, the expected utility is that of the ficticious
 simple OL
    exp_util =
 sum(norm_aggregate_prob_v.*multi_attribute_utilities_v);
end
```

Published with MATLAB® R2018b

```
function [dataset_m,new_dataset_m] =
 replace_values(dataset_m,rows_v,columns_v,new_values_v)
%replace_values searches through a dataset replacing values with those
 specified
%   Inputs: dataset_m        The dataset matrix which has values
 requiring
%                            replacement
%           rows_v           The vector of row indices for values in
 dataset_m
%                            which are to be replaced, must have equal
 length as
%                            columns_v and new_values_v
%           columns_v        The vector of column indices for values
 in dataset_m
%                            which are to be replaced, must have equal
 length as
%                            rows_v and new_values_v
%           new_values_v     The vector of new values for values in
 dataset_m
%                            which are to be replaced, must have equal
 length as
%                            rows_v and columns_v
%
%  Outputs: dataset_m        The dataset matrix which has values
 requiring
%                            replacement
%            new_dataset_m   dataset_m with values replaced as
 specified with
%                            indices in rows_v and columns_v with
 values in new_values_v
%

% CALCULATIONS:
    if (length(rows_v) == length(columns_v))&&(length(columns_v) ==
 length(new_values_v))
        new_dataset_m = dataset_m;
        for i = 1:length(rows_v)
            new_dataset_m(rows_v(i),columns_v(i))=new_values_v(i);
        end
    else
        disp('Array lengths not equal, cannot replace values.');
        return
    end
end
```

Published with MATLAB® R2018b

```
function [Meas_Res_Sampleknw_v,policynw,policynw_str,...
    Meas_Res_SamplekPw_v,policyPw,policyPw_str] =
 select_policy_noisy(...
    x_v,Ppr_v,selection)
% select_policy_noisy takes in a measurement from user, implements
 error checking ...
% and runs a decision analysis to return the policy
% Performs decision analysis using decision_tree_pump to load pump-
related
% utilities
%
% Inputs:    x_v                 A 85 x 1 vector of measurements for
 classification using
%                                linear discriminant classifers
%            Ppr_v               A 1 x 8 vector of apriori
 probabilities used to
%                                perform the decision analysis
%           selection            An integer (1 ,2, 3 or 4) describing
 the decision
%                                context as follows:
%                                1 = Vessel is stationary at the
 wharf, no other machinery running
%                                in engine room
%                                2 = Vessel is stationary at the
 wharf, main engines are running
%                                in engine room
%                                3 = Vessel is slow steaming in the
 harbour and we can
%                                stop the pump if necessary
%                                4 = Vessel is slow steaming in the
 harbour and we cannot
%                                stop the pump because there is an
 emergency which it
%                                must be made available for
% Outputs:  Meas_Res_Sampleknw_v  A 1 x 8 vector of posterior
 probabilities calculated from
%                                linear discriminant analysis of x_v
%                     policynw    The policy number, and integer
 (1,2,3,4 or 5)
%                                representing the best possible
 policy, calculated using the maximum
%                                expected utility rule and non-
weighted
%                                prior double
%                                probabilities
%                   policynw_str  The corresponding description of the
 selected policy
%                                number
%            Meas_Res_SamplekPw_v  A 1 x 8 vector of posterior
 probabilities calculated from
%                                 linear discriminant analysis of
%                                 x_v and weighted by trainined_Ppr_v
```

```
%                   policyPw       The policy number, and integer
 (1,2,3,4 or 5)
%                                  representing the best possible
 policy, calculated using the maximum
%                                  expected utility rule and weighted
 prior
%                                  probabilities double
%
%                 policyPw_str     The corresponding description of
 the selected policy
%                                  number

% CALCULATIONS:
    if nargin < 3 %Clarify inputs



        selection = menu('Please select:','Vessel is at the wharf, no
 equipment running'...
            ,'Vessel is at the wharf, other machinery is running'...
            ,'Vessel is slow steaming in the harbour and it is not an
 emergency'...
            ,'Vessel is slow steaming in the harbour and it is an
 emergency');

    end
    if nargin < 2 %Clarify inputs
        while (length(Ppr_v) ~=8)
            Ppr_v = input('Please re-enter prior probabilities as an 1
 x 8 vector in the form [a,b,c...].\n');
        end
        while (rem(selection,1) ~= 0)|| (selection <1) || (selection
>4)
            disp(compose("Please select a value from the following
 four options:\n1 = Vessel as at the wharf, no equipment running \n2
 = Vessel is at the wharf, other machinery is running \n3 = Vessel is
 slow steaming in the harbour and it is not an emergency\n4 = Vessel
 is slow steaming in the harbour and it is an emergency"));
            selection = input('Please enter a value.\n');
        end
    end
    %Check length of measurements and orientation of vector is correct
    while size(x_v,1)~=85
       x_v = input('Please re-enter measurement as an 85 x 1 vector in
 the form [a,b,c...]''.\n');
    end
    % Change selection into flag_dat and emergency variables
    flag_dat = selection;
    % Load previous prior probabilities
    load_filename_str=['DATA' int2str(flag_dat) '\trained_Ppr'];
    load(load_filename_str,'trained_Ppr_v');
    % Check if they are equal to the input Ppr_v
    if isequal(trained_Ppr_v,Ppr_v) == 1
```

```
        select_priors = 'Y';
    elseif isequal(trained_Ppr_v,Ppr_v) == 0
        select_priors = input('Trained priors not equal to input - Use
trained prior probability values? Y/N \n','s');
    end
    % Return if priors not equal and defaults not used
    if (select_priors == 'N') || (select_priors == 'n')
        policynw = 0;
        policynw_str = "N/A - Terminated operation as priors not
trained.";
        policyPw = 0;
        policyPw_str = "N/A - Terminated operation as priors not
trained.";
        input('Terminated operation as priors not trained. Returning
to calculation sub-menu. Press any key to continue.');
        return
    elseif (select_priors == 'Y') || (select_priors == 'y')
        disp("Using trained priors equal to input - proceeding to
calculate policy.");
    end
    % Classify and perform decision analysis
    %Classify the sample using trained classifiers developed with
trained priors
    [Meas_Res_Sampleknw_v,Meas_Res_SamplekPw_v]=
bayesian_classifier_measurement(flag_dat,x_v);
    %Calculate the policy
    [policynw,~,policyPw,~] =
decision_tree_pump(flag_dat,Meas_Res_Sampleknw_v,Meas_Res_SamplekPw_v);
    % Return the string description of the policies
    if (selection == 1) || (selection == 2)
        load('actions_atwharf.mat','actions_atwharf_c');
        policynw_str = actions_atwharf_c{policynw};
        policyPw_str = actions_atwharf_c{policyPw};
    elseif (selection == 3) || (selection == 4)
        load('actions_atsea.mat','actions_atsea_c');
        policynw_str = actions_atsea_c{policynw};
        policyPw_str = actions_atsea_c{policyPw};
    end
end
```

Published with MATLAB® R2018b

```
function [Meas_Res_Sampleknw_v,policynw,policynw_str,...
    Meas_Res_SamplekPw_v,policyPw,policyPw_str] =
 select_policy_pump(...
    x_v,Ppr_v,selection)
%select_policy_pump takes in a measurement from user, implements
 error ...
% checking and runs a decision analysis to return the policy
% Performs decision analysis using decision_tree_pump to load pump-
related
% utilities
%
% Inputs:    x_v                  A 85 x 1 vector of measurements for
 classification using
%                                 linear discriminant classifers
%            Ppr_v                A 1 x 8 vector of apriori
 probabilities used to
%                                 perform the decision analysis
%           selection             An integer (1 ,2, 3 or 4) describing
 the decision
%                                 context as follows:
%                                 1 = Vessel is stationary at the
 wharf, no other machinery running
%                                 in engine room
%                                 2 = Vessel is stationary at the
 wharf, main engines are running
%                                 in engine room
%                                 3 = Vessel is slow steaming in the
 harbour and we can
%                                 stop the pump if necessary
%                                 4 = Vessel is slow steaming in the
 harbour and we cannot
%                                 stop the pump because there is an
 emergency which it
%                                 must be made available for
% Outputs:  Meas_Res_Sampleknw_v  A 1 x 8 vector of posterior
 probabilities calculated from
%                                 linear discriminant analysis of x_v
%                     policynw    The policy number, and integer
 (1,2,3,4 or 5)
%                                 representing the best possible
 policy, calculated using the maximum
%                                 expected utility rule and non-
weighted
%                                 prior double
%                                 probabilities
%                   policynw_str  The corresponding description of the
 selected policy
%                                 number
%            Meas_Res_SamplekPw_v  A 1 x 8 vector of posterior
 probabilities
%                                  calculated from linear discriminant
 analysis of
```

```
%                                   x_v and weighted by trainined_Ppr_v
%                   policyPw        The policy number, and integer
 (1,2,3,4 or 5)
%                                   representing the best possible
 policy, calculated using the maximum
%                                   expected utility rule and weighted
 prior
%                                   probabilities double
%
%                 policyPw_str      The corresponding description of
 the selected policy
%                                   number

% CALCULATIONS:
    if nargin < 3 %Clarify inputs
        % Test the variables
        selection = menu('Please select:','Vessel is at the wharf, no
 equipment running'...
            ,'Vessel is at the wharf, other machinery is running'...
            ,'Vessel is slow steaming in the harbour and it is not an
 emergency'...
            ,'Vessel is slow steaming in the harbour and it is an
 emergency');
    end
    if nargin < 2 %Clarify inputs
        while (length(Ppr_v) ~=8)
            Ppr_v = input('Please re-enter prior probabilities as an 1
 x 8 vector in the form [a,b,c...].\n');
        end
        while (rem(selection,1) ~= 0)|| (selection <1) || (selection
 >4)
            disp(compose(...
                "Please select a value from the following four
 options:\n1 = Vessel as at the wharf, no equipment running \n2 =
 Vessel is at the wharf, other machinery is running \n3 = Vessel is
 slow steaming in the harbour and it is not an emergency\n4 = Vessel
 is slow steaming in the harbour and it is an emergency"));
            selection = input('Please enter a value.\n');
        end
    end
    %Check length of measurements and orientation of vector is correct
    while size(x_v,1)~=85
       x_v = input('Please re-enter measurement as an 85 x 1 vector in
 the form [a,b,c...]''.\n');
    end
    % Change selection into flag_dat and emergency variables
    flag_dat = selection;
    % Load previous prior probabilities
    load_filename_str=['DATA' int2str(flag_dat) '\trained_Ppr'];
    load(load_filename_str,'trained_Ppr_v');
    % Check if they are equal to the input Ppr_v
    if isequal(trained_Ppr_v,Ppr_v) == 1
        select_priors = 'Y';
    elseif isequal(trained_Ppr_v,Ppr_v) == 0
```

```
        select_priors = input(...
            'Trained priors not equal to input - Use trained prior
probability values? Y/N \n','s');
    end
    % Return if priors not equal and defaults not used
    if (select_priors == 'N') || (select_priors == 'n')
        policynw = 0;
        policynw_str = "N/A - Terminated operation as priors not
trained.";
        policyPw = 0;
        policyPw_str = "N/A - Terminated operation as priors not
trained.";
        input(...
            'Terminated operation as priors not trained. Returning to
calculation sub-menu. Press any key to continue.');
        return
    elseif (select_priors == 'Y') || (select_priors == 'y')
        disp("Using trained priors equal to input - proceeding to
calculate policy.");
    end
    % Classify and perform decision analysis
    %Classify the sample using trained classifiers developed with
trained priors
    [Meas_Res_Sampleknw_v,Meas_Res_SamplekPw_v]=
bayesian_classifier_measurement(flag_dat,x_v);
    %Calculate the policy
    [policynw,~,policyPw,~] =
decision_tree_pump(flag_dat,Meas_Res_Sampleknw_v,Meas_Res_SamplekPw_v);
    % Return the string description of the policies
    if (selection == 1) || (selection == 2)
        load('actions_atwharf.mat','actions_atwharf_c');
        policynw_str = actions_atwharf_c{policynw};
        policyPw_str = actions_atwharf_c{policyPw};
    elseif (selection == 3) || (selection == 4)
        load('actions_atsea.mat','actions_atsea_c');
        policynw_str = actions_atsea_c{policynw};
        policyPw_str = actions_atsea_c{policyPw};
    end
end
```

Published with MATLAB® R2018b

```
function [Meas_Res_Sampleknw_v,policynw,policynw_str,...
    Meas_Res_SamplekPw_v,policyPw,policyPw_str] =
 select_policy_test(...
    x_v,Ppr_v,selection)
%select_policy_test takes in a measurement from user, implements
 error ...
% checking and runs a decision analysis to return the policy
% Performs decision analysis using decision_tree_pump to load test
% utilities
%
% Inputs:    x_v                 A 85 x 1 vector of measurements for
 classification using
%                                 linear discriminant classifers
%            Ppr_v               A 1 x 8 vector of apriori
 probabilities used to
%                                 perform the decision analysis
%           selection            An integer (1 ,2, 3 or 4) describing
 the decision
%                                 context as follows:
%                                 1 = Vessel is stationary at the
 wharf, no other machinery running
%                                 in engine room
%                                 2 = Vessel is stationary at the
 wharf, main engines are running
%                                 in engine room
%                                 3 = Vessel is slow steaming in the
 harbour and we can
%                                 stop the pump if necessary
%                                 4 = Vessel is slow steaming in the
 harbour and we cannot
%                                 stop the pump because there is an
 emergency which it
%                                 must be made available for
% Outputs:  Meas_Res_Sampleknw_v  A 1 x 8 vector of posterior
 probabilities calculated from
%                                 linear discriminant analysis of x_v
%                     policynw    The policy number, and integer
 (1,2,3,4 or 5)
%                                 representing the best possible
 policy, calculated using the maximum
%                                 expected utility rule and non-
weighted
%                                 prior double
%                                 probabilities
%                   policynw_str  The corresponding description of the
 selected policy
%                                 number
%            Meas_Res_SamplekPw_v  A 1 x 8 vector of posterior
 probabilities calculated from linear discriminant analysis of
%                                  x_v and weighted by trainined_Ppr_v
%                   policyPw       The policy number, and integer
 (1,2,3,4 or 5)
```

```
%                                   representing the best possible
 policy, calculated using the maximum
%                                   expected utility rule and weighted
 prior
%                                   probabilities double
%
%                 policyPw_str      The corresponding description of
 the selected policy
%                                   number

% CALCULATIONS:
    if nargin < 3 %Clarify inputs
        % Test the variables
        selection = menu('Please select:','Vessel is at the wharf, no
 equipment running'...
            ,'Vessel is at the wharf, other machinery is running'...
            ,'Vessel is slow steaming in the harbour and it is not an
 emergency'...
            ,'Vessel is slow steaming in the harbour and it is an
 emergency');
    end
    if nargin < 2 %Clarify inputs
        while (length(Ppr_v) ~=8)
            Ppr_v = input('Please re-enter prior probabilities as an 1
 x 8 vector in the form [a,b,c...].\n');
        end
        while (rem(selection,1) ~= 0)|| (selection <1) || (selection
 >4)
            disp(compose("Please select a value from the following
 four options:\n1 = Vessel as at the wharf, no equipment running \n2
 = Vessel is at the wharf, other machinery is running \n3 = Vessel is
 slow steaming in the harbour and it is not an emergency\n4 = Vessel
 is slow steaming in the harbour and it is an emergency"));
            selection = input('Please enter a value.\n');
        end
    end
    %Check length of measurements and orientation of vector is correct
    while size(x_v,1)~=85
       x_v = input('Please re-enter measurement as an 85 x 1 vector in
 the form [a,b,c...]''.\n');
    end
    % Change selection into flag_dat and emergency variables
    flag_dat = selection;
    % Load previous prior probabilities
    load_filename_str=['DATA' int2str(flag_dat) '\trained_Ppr'];
    load(load_filename_str,'trained_Ppr_v');
    % Check if they are equal to the input Ppr_v
    if isequal(trained_Ppr_v,Ppr_v) == 1
        select_priors = 'Y';
    elseif isequal(trained_Ppr_v,Ppr_v) == 0
        select_priors = input('Trained priors not equal to input - Use
 trained prior probability values? Y/N \n','s');
    end
    % Return if priors not equal and defaults not used
```

```
    if (select_priors == 'N') || (select_priors == 'n')
        policynw = 0;
        policynw_str = "N/A - Terminated operation as priors not
trained.";
        policyPw = 0;
        policyPw_str = "N/A - Terminated operation as priors not
trained.";
        input('Terminated operation as priors not trained. Returning
to calculation sub-menu. Press any key to continue.');
        return
    elseif (select_priors == 'Y') || (select_priors == 'y')
        disp("Using trained priors equal to input - proceeding to
calculate policy.");
    end
    % Classify and perform decision analysis
    %Classify the sample using trained classifiers developed with
trained priors
    [Meas_Res_Sampleknw_v,Meas_Res_SamplekPw_v]=
bayesian_classifier_measurement(flag_dat,x_v);
    %Calculate the policy
    [policynw,~,policyPw,~] =
decision_tree_test(flag_dat,Meas_Res_Sampleknw_v,Meas_Res_SamplekPw_v);
    % Return the string description of the policies
    if (selection == 1) || (selection == 2)
        load('actions_atwharf.mat','actions_atwharf_c');
        policynw_str = actions_atwharf_c{policynw};
        policyPw_str = actions_atwharf_c{policyPw};
    elseif (selection == 3) || (selection == 4)
        load('actions_atsea.mat','actions_atsea_c');
        policynw_str = actions_atsea_c{policynw};
        policyPw_str = actions_atsea_c{policyPw};
    end
end
```

Published with MATLAB® R2018b

```
function [exp_util]=simple_lottery(Pos_v,multi_attribute_utilities_v)
% simple_lottery calculates the expected utility of a simple lottery
% Inputs must have same length
%
% Inputs: Pos_v                       1x8 vector of state posterior
 probabilities
%         multi_attribute_utilities_v 1x8 vector of multi attribute
 utilities
%
% Output: exp_util                    Expected utility of the lottery
 double

% CALCULATIONS:
    exp_util = sum(Pos_v.*multi_attribute_utilities_v);
end
```

Published with MATLAB® R2018b

```
function []=subj_prob_database(selection)
% subj_prob_database manages the pump measurements and test
 measurments for classification
% This function loads all pump subjective probabilities stored within
 the system,
% enabling the user to view and modify them
%
%   Note on the datasets:
%            DATA(flag_dat)\subjective_probability_default.mat matrix
 of 8 row x 5
%            column matrix of subjective probabilities surveyed from
 an expert regarding a
%            shipboard pump. This is not overwritten by the
%            software. For the physical meaning of these data, refer
 to
%            Thesis/LDA paper about the pump. We load and modify the
%            worst-case values, recalculating the best-case values
 from the
%            modifications prior to saving.
%
%            DATA(flag_dat)\subjective_probability_testdata.mat is a
 randomly generated matrix
%            of 85 features x 100 samples which can be regenerated for
 testing purposes
%
%            In all cases we save modifications to files as:
%            DATA(flag_dat)\subjective_probability.mat
%
% Input:  selection An integer (1 ,2, 3 or 4) describing the decision
%                   context as follows:
%                   1 = Vessel is stationary at the wharf, no other
 machinery running
%                       in engine room
%                   2 = Vessel is stationary at the wharf, main
 engines are running
%                       in engine room
%                   3 = Vessel is slow steaming in the harbour and we
 can
%                       stop the pump if necessary
%                   4 = Vessel is slow steaming in the harbour and we
 cannot
%                       stop the pump because there is an emergency
 which it
%                       must be made available for

% CALCULATIONS:
    % Change selection into flag_dat and emergency variables
    flag_dat = selection;
    % String to describe data selected by user
    working_dataset_c ={'Vessel is at the wharf, no equipment
 running'...
        ,'Vessel is at the wharf, other machinery is running'...
```

```
       ,'Vessel is slow steaming in the harbour and it is not an
emergency'...
       ,'Vessel is slow steaming in the harbour and it is an
emergency'};
   choice = 0;
   while choice ~=10
       disp(strcat('Selected dataset
describes:',working_dataset_c{flag_dat}));
       choice = menu('Subjective Probability Database Options',...
           '1 = View Pump Subjective Probabilities',...
           '2 = Modify Pump Subjective Probability Values',...
           '3 = Add/remove Pump Subjective Probabilities',...
           '4 = Restore Pump Data to Default',...
           '5 = View Test Subjective Probabilities',...
           '6 = Modify Test Subjective Probability Values',...
           '7 = Add/remove Test Subjective Probabilities',...
           '8 = Restore Test Data to Default',...
           '9 = Change datasets',...
           '10 = Return to Main Menu');
       % Load and manage the data according to menu choices
       if (choice == 1) %View only
           view_again = 'Y';
           inp_filename1_str=['DATA'
int2str(flag_dat) '\subjective_probability'];

load(inp_filename1_str,'subjective_probability_worstcase_m');
           while (view_again == 'Y') || (view_again == 'y')
               disp(strcat('Data loaded into workspace. Dataset size
is ...',...

num2str(size(subjective_probability_worstcase_m,1)),'... features
by ...',...

num2str(size(subjective_probability_worstcase_m,2)),'... samples.'));
               rows_v = input('Please input row numbers to view as a
vector[a,b].');
               columns_v = input('Please input column numbers to view
as a vector[a,b].');
               disp(...
                   'Displaying specified range and pausing execution.
All other values displayed as zero. Press any key to continue.');
               [~,new_dataset_m] = display_values(...

subjective_probability_worstcase_m,rows_v,columns_v);
               disp(new_dataset_m);
               pause;
               view_again = input('View another range? Y/N as
character');
           end
       elseif (choice == 2) %%View and modify values
           % View data as per previous choice
           inp_filename1_str=['DATA'
int2str(flag_dat) '\subjective_probability'];
           modify_data = 'Y';
```

```
            while (modify_data == 'Y') || (modify_data == 'y') %Modify
data

load(inp_filename1_str,'subjective_probability_worstcase_m');
                disp(strcat('Data loaded into workspace. Dataset size
is ...',...

num2str(size(subjective_probability_worstcase_m,1)),'... features
by ...',...

num2str(size(subjective_probability_worstcase_m,2)),'... samples.'));
                modify_indices_m = input(...
                    'Please input data rows and columns
to modify as well as new values in a matrix of the form
[row1,col1,newval1;row2,col2,newval2...].');
                disp(modify_indices_m);
                disp('Modifying values then pausing execution. Press
any key to continue.');
                [~,new_dataset_m] = replace_values(...

subjective_probability_worstcase_m,modify_indices_m(:,1),...
                    modify_indices_m(:,2),modify_indices_m(:,3));
                disp('Old dataset:');
                disp(subjective_probability_worstcase_m);
                disp('New dataset with replacements:');
                disp(new_dataset_m);
                pause;
                %Prompt to save changes or discard
                update_data = input('Save changes? Y/N as character');
                if (update_data == 'Y') || (update_data == 'y')
                    subjective_probability_worstcase_m=new_dataset_m;
                    subjective_probability_bestcase_m = 1-
subjective_probability_worstcase_m;
                    out_filename1_str=['DATA'
int2str(flag_dat) '\subjective_probability'];

save(out_filename1_str,'subjective_probability_worstcase_m',...
                        'subjective_probability_bestcase_m');
                    input('Changes saved. Press any key to
continue.');
                elseif (update_data == 'N') || (update_data == 'n')
                    input('Changes NOT saved. Press any key to
continue.');
                end
                modify_data = input('Modify another range? Y/N as
character');
            end
        elseif (choice == 3) %Append or reduce dataset
            resize_dataset = 'Y';
            view_again = 'Y';
            inp_filename1_str=['DATA'
int2str(flag_dat) '\subjective_probability'];

load(inp_filename1_str,'subjective_probability_worstcase_m');
```

```
            while (view_again == 'Y') || (view_again == 'y')
                disp(strcat('Data loaded into workspace. Dataset size
is ...',...

num2str(size(subjective_probability_worstcase_m,1)),'... features
by ...',...

num2str(size(subjective_probability_worstcase_m,2)),'... samples.'));
                rows_v = input('Please input row numbers to view as a
vector[a,b].');
                columns_v = input('Please input column numbers to view
as a vector[a,b].');
                disp(...
                    'Displaying specified range and pausing execution.
All other values displayed as zero. Press any key to continue.');
                [~,new_dataset_m] = display_values(...

subjective_probability_worstcase_m,rows_v,columns_v);
                disp(new_dataset_m);
                pause;
                view_again = input('View another range? Y/N as
character');
            end
            disp('Proceeding to change size of dataset ...');
            resized_dataset_m = subjective_probability_worstcase_m;
            while (resize_dataset == 'Y') || (resize_dataset == 'y')
                num_rows = size(resized_dataset_m,1);
                num_columns = size(resized_dataset_m,2);
                disp(...
                    strcat('Data loaded into workspace. Dataset size
is ...',num2str(num_rows)...
                    ,'... features by ...',...
                    num2str(num_columns),'... samples.'));
                disp('Warning: Changing size from default will prevent
calculations using current decision analysis ...');
                action = menu('What do you want to do?','1 = Append
dataset',...
                    '2 = Extract subset of dataset');
                if action == 1
                    disp('Appending data by measurement to dataset.');
                    append_prompt_str = strcat('Please enter a matrix
of ...',...
                        num2str(num_rows),'... rows and desired number
of columns in the form [a,b,c...]');
                    append_measurements_m = input(append_prompt_str);
                    resized_dataset_results_m = [resized_dataset_m,...
                        append_measurements_m];
                    new_rows = size(resized_dataset_results_m,1);
                    new_columns = size(resized_dataset_results_m,2);
                elseif action == 2
                    disp('Extracting subset of dataset.');
                    extract_rows_v = input(...
                        'Please enter rows of dataset to extract as a
vector of the form [a,b,c...].');
```

```
                    extract_rows_v = sort(extract_rows_v);
                    extract_columns_v = input(...
                        'Please enter columns of dataset to extract as
 a vector of the form [a,b,c...].');
                    extract_columns_v = sort(extract_columns_v);
                    resized_dataset_results_m = ...

 resized_dataset_m(extract_rows_v,extract_columns_v);
                    new_rows = size(resized_dataset_results_m,1);
                    new_columns = size(resized_dataset_results_m,2);
                end
                disp(strcat('The resulting dataset is a matrix
 with ...',...
                    num2str(new_rows),'... rows
 and ...',num2str(new_columns),':'));
                disp(resized_dataset_results_m);
                disp('Press any key to continue.');
                pause;
                resize_dataset = input('Resize dataset further? Y/N as
 character');
            end
            save_resize = 'N'; %#ok<NASGU>
            disp('Warning: Changing size from default will prevent
 calculations using current decision analysis ...');
            save_resize = input('Save changes and overwrite dataset?
 Y/N as character');
            if (save_resize == 'Y') || (save_resize == 'y')
                subjective_probability_worstcase_m =
 resized_dataset_results_m;
                out_filename2_str=['DATA'
 int2str(flag_dat) '\subjective_probability'];
                subjective_probability_bestcase_m = 1-
subjective_probability_worstcase_m;

 save(out_filename2_str,'subjective_probability_worstcase_m',...
                    'subjective_probability_bestcase_m');
                input('Resized data saved. Press any key to
 continue.');
            elseif (save_resize == 'N') || (save_resize == 'n')
                input('Changes not saved. Press any key to
 continue.');
            end
        elseif (choice == 4)
            inp_filename3_str=['DATA'
 int2str(flag_dat) '\subjective_probability_default'];

 load(inp_filename3_str,'subjective_probability_worstcase_default_m');
            subjective_probability_worstcase_m =
 subjective_probability_worstcase_default_m;
            subjective_probability_bestcase_m = 1-
subjective_probability_worstcase_m;
            out_filename2_str=['DATA'
 int2str(flag_dat) '\subjective_probability'];
```

```
save(out_filename2_str,'subjective_probability_worstcase_m','subjective_probabili
           input('Default data restored. Press any key to
continue.');
       elseif (choice == 5) %View only
           view_again = 'Y';
           inp_filename1_str=['DATA'
int2str(flag_dat) '\subjective_probability_testdata'];

load(inp_filename1_str,'subjective_probability_worstcase_testdata_m');
           while (view_again == 'Y') || (view_again == 'y')
               disp(strcat('Data loaded into workspace. Dataset size
is ...',...

num2str(size(subjective_probability_worstcase_testdata_m,1)),'...
features by ...',...

num2str(size(subjective_probability_worstcase_testdata_m,2)),'...
samples.'));
               rows_v = input('Please input row numbers to view as a
vector[a,b].');
               columns_v = input('Please input column numbers to view
as a vector[a,b].');
               disp(...
                   'Displaying specified range and pausing execution.
All other values displayed as zero. Press any key to continue.');
               [~,new_dataset_m] = display_values(...

subjective_probability_worstcase_testdata_m,rows_v,columns_v);
               disp(new_dataset_m);
               pause;
               view_again = input('View another range? Y/N as
character');
           end
       elseif (choice == 6) %%View and modify
           % View data as per previous choice
           inp_filename1_str=['DATA'
int2str(flag_dat) '\subjective_probability_testdata'];
           modify_data = 'Y';
           while (modify_data == 'Y') || (modify_data == 'y') %Modify
data

load(inp_filename1_str,'subjective_probability_worstcase_testdata_m');
               disp(strcat('Data loaded into workspace. Dataset size
is ...',...

num2str(size(subjective_probability_worstcase_testdata_m,1)),'...
features by ...',...

num2str(size(subjective_probability_worstcase_testdata_m,2)),'...
samples.'));
               modify_indices_m = input(...
```

```
                    'Please input data rows and columns
to modify as well as new values in a matrix of the form
[row1,col1,newval1;row2,col2,newval2...].');
                disp(modify_indices_m);
                disp('Modifying values then pausing execution. Press
any key to continue.');
                [~,new_dataset_m] = replace_values(...

subjective_probability_worstcase_testdata_m,modify_indices_m(:,1),...
                    modify_indices_m(:,2),modify_indices_m(:,3));
                disp('Old dataset:');
                disp(subjective_probability_worstcase_testdata_m);
                disp('New dataset with replacements:');
                disp(new_dataset_m);
                pause;
                %Prompt to save changes or discard
                update_data = input('Save changes? Y/N as character');
                if (update_data == 'Y') || (update_data == 'y')

subjective_probability_worstcase_testdata_m=new_dataset_m;
                    out_filename1_str=['DATA' int2str(flag_dat)...
                        '\subjective_probability_testdata'];
                    subjective_probability_bestcase_testdata_m = 1 ...
                        -
subjective_probability_worstcase_testdata_m; %#ok<NASGU>
                    save(...

out_filename1_str,'subjective_probability_worstcase_testdata_m',...

 'subjective_probability_worstcase_testdata_m');
                    input('Changes saved. Press any key to
continue.');
                elseif (update_data == 'N') || (update_data == 'n')
                    input('Changes NOT saved. Press any key to
continue.');
                end
                modify_data = input('Modify another range? Y/N as
character');
            end
        elseif (choice == 7) %Append or reduce dataset
            resize_dataset = 'Y';
            view_again = 'Y';
            inp_filename1_str=['DATA'
int2str(flag_dat) '\subjective_probability_testdata'];

load(inp_filename1_str,'subjective_probability_worstcase_testdata_m');
            while (view_again == 'Y') || (view_again == 'y')
                disp(strcat('Data loaded into workspace. Dataset size
is ...',...

num2str(size(subjective_probability_worstcase_testdata_m,1)),'...
features by ...',...
```

```
num2str(size(subjective_probability_worstcase_testdata_m,2)),'...
samples.'));
                rows_v = input('Please input row numbers to view as a
vector[a,b].');
                columns_v = input('Please input column numbers to view
as a vector[a,b].');
                disp(...
                    'Displaying specified range and pausing execution.
All other values displayed as zero. Press any key to continue.');
                [~,new_dataset_m] = display_values(...

subjective_probability_worstcase_testdata_m,rows_v,columns_v);
                disp(new_dataset_m);
                pause;
                view_again = input('View another range? Y/N as
character');
            end
            disp('Proceeding to change size of dataset ...');
            resized_dataset_m =
subjective_probability_worstcase_testdata_m;
            while (resize_dataset == 'Y') || (resize_dataset == 'y')
                num_rows = size(resized_dataset_m,1);
                num_columns = size(resized_dataset_m,2);
                disp(strcat('Data loaded into workspace. Dataset size
is ...',...
                    num2str(num_rows),'... features by ...',...
                    num2str(num_columns),'... samples.'));
                disp('Warning: Changing size from default will prevent
calculations using current decision analysis ...');
                action = menu('What do you want to do?',...
                    '1 = Append dataset','2 = Extract subset of
dataset');
                if action == 1
                    disp('Appending data by measurement to dataset.');
                    append_prompt_str = strcat(...
                        'Please enter a matrix
of ...',num2str(num_rows),...
                        '... rows and desired number of columns in the
form [a,b,c...]');
                    append_measurements_m = input(append_prompt_str);
                    resized_dataset_results_m =
[resized_dataset_m,append_measurements_m];
                    new_rows = size(resized_dataset_results_m,1);
                    new_columns = size(resized_dataset_results_m,2);
                elseif action == 2
                    disp('Extracting subset of dataset.');
                    extract_rows_v = input(...
                        'Please enter rows of dataset to extract as a
vector of the form [a,b,c...].');
                    extract_rows_v = sort(extract_rows_v);
                    extract_columns_v = input(...
                        'Please enter columns of dataset to extract as
a vector of the form [a,b,c...].');
```

```
                    extract_columns_v = sort(extract_columns_v);
                    resized_dataset_results_m =
resized_dataset_m(extract_rows_v,extract_columns_v);
                    new_rows = size(resized_dataset_results_m,1);
                    new_columns = size(resized_dataset_results_m,2);
                end
                disp(strcat('The resulting dataset is a matrix
with ...',...
                    num2str(new_rows),'... rows
and ...',num2str(new_columns),':'));
                disp(resized_dataset_results_m);
                disp('Press any key to continue.');
                pause;
                resize_dataset = input('Resize dataset further? Y/N as
character');
            end
            save_resize = 'N'; %#ok<NASGU>
            disp('Warning: Changing size from default will prevent
calculations using current decision analysis ...');
            save_resize = input('Save changes and overwrite dataset?
Y/N as character');
            if (save_resize == 'Y') || (save_resize == 'y')
                subjective_probability_worstcase_testdata_m =
resized_dataset_results_m;
                out_filename2_str=['DATA'
int2str(flag_dat) '\subjective_probability_testdata'];
                subjective_probability_bestcase_testdata_m = 1 - ...
                    subjective_probability_worstcase_testdata_m;
                save(out_filename2_str,...
                    'subjective_probability_worstcase_testdata_m',...
                    'subjective_probability_bestcase_testdata_m');
                input('Resized data saved. Press any key to
continue.');
            elseif (save_resize == 'N') || (save_resize == 'n')
                input('Changes not saved. Press any key to
continue.');
            end
        elseif (choice == 8)
            inp_filename3_str=['DATA'
int2str(flag_dat) '\subjective_probability_testdata_default'];

load(inp_filename3_str,'subjective_probability_worstcase_testdata_default_m');
            subjective_probability_worstcase_testdata_m =
subjective_probability_worstcase_testdata_default_m;
            subjective_probability_bestcase_testdata_m = 1 -
subjective_probability_worstcase_testdata_m;
            out_filename2_str=['DATA'
int2str(flag_dat) '\subjective_probability_testdata'];
            save(out_filename2_str,...

 'subjective_probability_worstcase_testdata_m','subjective_probability_bestcase_t
            input('Default data restored. Press any key to
continue.');
        elseif (choice == 9)
```

```
            %Prompt user for a new choice
            selection = menu('Please select dataset of interest:',...
                'Vessel is at the wharf, no equipment running'...
                ,'Vessel is at the wharf, other machinery is
 running'...
                ,'Vessel is slow steaming in the harbour and it is not
 an emergency'...
                ,'Vessel is slow steaming in the harbour and it is an
 emergency');
            % Change selection into flag_dat and emergency variables
            flag_dat = selection;
        end
    end
    input('Returning to Main Menu. Press any key to continue.');
end
```

Published with MATLAB® R2018b

```
function
 [u,flag_over,flagstop,str]=universal_utility_dec(x,a,x0,xd,xu)
% UNIVERSAL_UTILITY_DEC finds the utility values corresponding to
 given
%    prizes, using the universal decreasing arctangent utility
 function:
%u(x)=[arctg(a.xu-a.x0)-arctg(a.x-a.x0)]/[arctg(a.xu-a.x0)-arctg(a.xd-
a.x0)]
%
% [u,flag_over,flagstop,str]=universal_utility_dec(x,a,x0,xd,xu)
%
% x         an arbitrary 2D matrix of possible values of the prize in
 the
%           interval [xd;xu]
% a         the value of the positive parameter a in the function
% x0        the value of the parameter x0 in the function
% xd        the lower boundary of the prize interval, where u(xd)=1
% xu        the upper boundary of the prize interval, where u(xu)=0
%
% u         a matrix of utility values, corresponding to the prize
%           values in x
% flag_over flag for overflow when flag_over=1, showing linear utility
% flagstop  flag for abnormal termination of 'universal_utility_dec'
 if
%           flagstop~=0
% str       text string containing a message from
 'universal_utility_dec'
%           if flagstop~=0
%
%      See also universal_utility_inv_dec
h_universal_utility_dec_ic=@universal_utility_dec_ic;
er=0;
if nargin<5
    er=1;
    flagstop=6;
    str='Incorrect number of input parameters (message from
 ''universal_utility_dec'')';
end
if er==0

 [x,a,x0,xd,xu,flagstop,str,er]=h_universal_utility_dec_ic(x,a,x0,xd,xu);
end
if flagstop~=0
    u=deal([]);
    error(sprintf('%s\ninternal error code er=%i',str,er));
    return
end
uru=atan(a*(xu-x0));
urd=atan(a*(xd-x0));
if abs(urd-uru)>100*eps
    u=(uru-atan(a*(x-x0)))/(uru-urd);
    flag_over=0;
```

```
else
    u=interp1([xd,xu],[1,0],x);
    flag_over=1;
end
end

function
 [x,a,x0,xd,xu,flagstop,str,er]=universal_utility_dec_ic(x,a,x0,xd,xu)
flagstop=0;
str='';
er=0;
if ndims(x)~=2
    er=1;
elseif size(x,1)==0 | size(x,2)==0
    er=2;
end
if er==0
    er=chcknmat(x,size(x,1),size(x,2));
end
if er~=0
    flagstop=1;
    str='Incorrect input parameter x (message from
 ''universal_utility_dec'')';
    [x,a,x0,xd,xu]=deal([]);
    return
end
er=0;
er=chcknmat(a,1,1);
if er==0 & a<=0
    er=1;
end
if er~=0
    flagstop=2;
    str='Incorrect input parameter a (message from
 ''universal_utility_dec'')';
    [x,a,x0,xd,xu]=deal([]);
    return
end
er=0;
er=chcknmat(x0,1,1);
if er~=0
    flagstop=3;
    str='Incorrect input parameter x0 (message from
 ''universal_utility_dec'')';
    [x,a,x0,xd,xu]=deal([]);
    return
end
er=0;
er=chcknmat(xd,1,1);
if er==0 & any(any(x<xd))
    er=10;
end
if er~=0
    flagstop=4;
```

```
    str='Incorrect input parameter xd (message from
 ''universal_utility_dec'')';
    [x,a,x0,xd,xu]=deal([]);
    return
end
er=0;
er=chcknmat(xu,1,1);
if er==0 & xd>=xu
    er=10;
end
if er==0 & any(any(x>xu))
    er=11;
end
if er~=0
    flagstop=5;
    str='Incorrect input parameter xu (message from
 ''universal_utility_dec'')';
    [x,a,x0,xd,xu]=deal([]);
    return
end
end
```

Published with MATLAB® R2018b

```
function
 [x,ang,flag_over,flagstop,str]=universal_utility_inv_dec(u,a,x0,xd,xu)
% UNIVERSAL_UTILITY_INV_DEC finds prizes corresponding to given
 utility
%     values, using the inverse of the universal decreasing arctangent
%     utility function:
% x(u)=tan[(1-u)*atan(a.xu-a.x0))+u*atan(a.xd-a.x0)]/a+x0
%
%
 [x,ang,flag_over,flagstop,str]=universal_utility_inv_dec(u,a,x0,xd,xu)
%
% u         an arbitrary 2D matrix of possible utility values in the
%           interval [0, 1]
% a         the value of the positive parameter a in the function
% x0        the value of the parameter x0 in the function
% xd        the lower boundary of the prize interval, where x(0)=xu
% xu        the upper boundary of the prize interval, where x(1)=xd
%
% x         a matrix of prize values, corresponding to the utility
%           values in u
% ang       a matrix of auxiliary angles: (1-u)*atan(a.xu-
a.x0))+u*atan(a.xd-a.x0)
% flag_over flag for overflow when flag_over=1, showing linear utility
% flagstop  flag for abnormal termination of
 'universal_utility_inv_dec' if
%           flagstop~=0
% str       text string containing a message from
 'universal_utility_inv_dec'
%           if flagstop~=0
%
%      See also universal_utility
h_universal_utility_inv_dec_ic=@universal_utility_inv_dec_ic;
er=0;
if nargin<5
    er=1;
    flagstop=6;
    str='Incorrect number of input parameters (message from
 ''universal_utility_inv_dec'')';
end
if er==0

 [u,a,x0,xd,xu,flagstop,str,er]=h_universal_utility_inv_dec_ic(u,a,x0,xd,xu);
end
if flagstop~=0
    x=deal([]);
    error(sprintf('%s\ninternal error code er=%i',str,er));
    return
end
uru=atan(a*(xu-x0));
urd=atan(a*(xd-x0));
if abs(urd-uru)>100*eps
    ang=atan(a*(xu-x0))+u*(atan(a*(xd-x0))-atan(a*(xu-x0)));
```

```
    x=tan(ang)/a+x0;
    flag_over=0;
else
    x=interp1([1,0],[xd,xu],u);
    ang=[];
    flag_over=1;
end
end

function
 [u,a,x0,xd,xu,flagstop,str,er]=universal_utility_inv_dec_ic(u,a,x0,xd,xu)
flagstop=0;
str='';
er=0;
if ndims(u)~=2
    er=1;
elseif size(u,1)==0 | size(u,2)==0
    er=2;
end
if er==0
    er=chcknmat(u,size(u,1),size(u,2));
end
if er==0 & (any(any(u<0)) | any(any(u>1)))
    er=3;
end
if er~=0
    flagstop=1;
    str='Incorrect input parameter u (message from
 ''universal_utility_inv_dec'')';
    [u,a,x0,xd,xu]=deal([]);
    return
end
er=0;
er=chcknmat(a,1,1);
if er==0 & a<=0
    er=1;
end
if er~=0
    flagstop=2;
    str='Incorrect input parameter a (message from
 ''universal_utility_inv_dec'')';
    [u,a,x0,xd,xu]=deal([]);
    return
end
er=0;
er=chcknmat(x0,1,1);
if er~=0
    flagstop=3;
    str='Incorrect input parameter x0 (message from
 ''universal_utility_inv_dec'')';
    [u,a,x0,xd,xu]=deal([]);
    return
end
er=0;
```

```
er=chcknmat(xd,1,1);
if er~=0
    flagstop=4;
    str='Incorrect input parameter xd (message from
 ''universal_utility_inv_dec'')';
    [u,a,x0,xd,xu]=deal([]);
    return
end
er=0;
er=chcknmat(xu,1,1);
if er==0 & xd>=xu
    er=10;
end
if er~=0
    flagstop=5;
    str='Incorrect input parameter xu (message from
 ''universal_utility_inv_dec'')';
    [u,a,x0,xd,xu]=deal([]);
    return
end
end
```

Published with MATLAB® R2018b

```
function []=util_rng_database()
% util_rng_database manages the surveyed utilities, ki weights
 test ...
% utilities and test ki weights for
% decision_analysis
%   This function loads all utility range values stored within the
 system,
%   enabling the user to view and modify them
%
%   Note on the datasets:
%            utils\survey_utils_default.mat contains:
%             attribute_names_c           A 1 x 6 cell array of the
 names of the
%                                         attributes (rows in
 utilities_matrix_m(:,2:end)) in developing
%                                         an MAU function for decision
 analyis
%             data_names_c                A 1 x 6 cell array of the
 names of the
%                                         values (columns in
 utilities_matrix_m(:,2:end))
%             utilities_matrix_m(:,2:end) A 35 x 6 matrix of data
 gathered for PhD project through
%                                         surveying an expert
%             ki_weights_m                A 6 x 2 matrix of of
 attribute weights corresponding to data
%                                         gathered for PhD project
 through surveying an expert, stored as
%                                         [ki(1)min , ki(1)max ; ...];
%
%           utils\survey_utils.mat contains same data as
%           survey_utils_default.mat originally, but functions
 overwrite
%           this file as the user specifies with new data
%
%           utils\test_utils_default.mat and utils\test_utils.mat are
%           analogies to survey-based datasets, but these can be used
 to
%           store another independent set of utilities to test the
 system
%
%           utils\constants_pump_default.mat contains constants used
 to
%           calculate attribute values before they are converted into
%           utilities. These were determined with the pump maintenance
 in
%           mind, having surveyed an expert accordingly
%
%           utils\constants_pump_default.mat contains same data as
%           survey_utils_default.mat originally, but functions
 overwrite
%           this file as the user specifies with new data
```

```
%
%           Lastly, utils\test_constants_default.mat and utils
\test_constants.mat are
%           analogies to survey-based datasets, but these can be used
 to
%           store another independent set of utilities to test the
 system
%
%           In all cases the files which can be modified by the user
 are
%           saved without the '_default' suffix. Defaults are not
%           overwritten.

% CALCULATIONS:
    choice = 0;
    while choice ~=17
        choice = menu('Utility Range Database Options:',...
            '1 = View Pump Utility Ranges and Refit Attribute Utility
 Function Parameters',...
            '2 = Modify Pump Utility Ranges Values',...
            '3 = Add/remove Pump Utility Range',...
            '4 = View Pump Attribute Weights (ki weights)',...
            '5 = Modify Pump Attribute Weights (ki weights)',...
            '6 = View Pump Calculations Constants',...
            '7 = Modify Pump Calculations Constants',...
            '8 = Restore Pump Data to Default',...
            '9 = View Test Utility Ranges and Refit Attribute Utility
 Function Parameters',...
            '10 = Modify Test Utility Ranges Values',...
            '11 = Add/remove Test Utility Ranges',...
            '12 = View Test Attribute Weights (ki weights)',...
            '13 = Modify Test Attribute Weights (ki weights)',...
            '14 = View Test Calculations Constants',...
            '15 = Modify Test Calculations Constants',...
            '16 = Restore Test Data to Default',...
            '17 = Return to Main Menu');
       % Load and manage the data according to menu choices
       if (choice == 1) %View only (also calculates optimal data
 fitted parameters)
            view_again = 'Y';
            inp_filename1_str=['utils' '\' 'survey_utils'];

 load(inp_filename1_str,'attribute_names_c','data_names_c',...
                'utilities_matrix_m','ki_weights_m');
            while (view_again == 'Y') || (view_again == 'y')
                disp(strcat('Data loaded into workspace. Dataset size
 is ...',...
                    num2str(size(utilities_matrix_m(:,2:end),1)),'...
 rows by ...',...
                    num2str(size(utilities_matrix_m(:,2:end),2)),'...
 columns.'));
                disp(strcat('This dataset is made up of utilities
 for ...',...
```

```
                    num2str(length(attribute_names_c)),'...
attributes'));
                attr = menu('Please select an attribute to view
associated data:',...
                    attribute_names_c);
                disp(strcat('Displaying utility ranges for
attribute: ...',...
                    attribute_names_c{attr}));
                subset_v = utilities_matrix_m(:,1)== attr;
                attribute_utilities_m =
utilities_matrix_m(subset_v,2:end);
                if (attr ~=4) && (attr ~=6)
                    attribute_utilities_t =
array2table(attribute_utilities_m,...
                        'VariableNames',data_names_c);
                    disp(attribute_utilities_t);
                    % Fit optimal parameters for data considering
uncertainty on x, all are descending
                    %Define contants and re-arrange data
                    disp('Optimising attribute function parameters and
plotting ...');
                    x_m = attribute_utilities_m(2:(end-1),2:3)';
                    x_m = sort(x_m,2); %#ok<*UDIM>
                    u_v = attribute_utilities_m(2:(end-1),end)';
                    u_v = sort(u_v,'descend');
                    abeg = 0.5;
                    x0beg = 2;
                    xd=attribute_utilities_m(end,2); %Lowest value
                    xu=attribute_utilities_m(1,3); %Highest value
                    flagparam = [1 1];
                    flagfile = 1;
                    flagplot = 1;
                    %Fit parameters
                    [aopt,x0opt,HIopt,~,~,~]=optparam_xdec...

(abeg,x0beg,flagparam,x_m,u_v,xd,xu,flagplot,flagfile);
                    disp(strcat('Optimal attribute function parameters
are: ...',...
                        num2str(aopt),'...and...',num2str(x0opt),...
                        '... with a fitting error
of ...',num2str(HIopt)));
                    % Save fitting results for attribute from refresh
of
                    % parameters
                    inp_filename2 = 'optpar_xdec';

load(inp_filename2,'aopt','x0opt','HIopt','flagparam');
                    out_filename2 = ['utils' '\' 'pump_attr_funct_'
num2str(attr) '_params'];

save(out_filename2,'aopt','x0opt','xd','xu','HIopt','flagparam');
                    disp(strcat('Attribute function parameters saved
as: ... ',out_filename2));
                    delete 'optpar_xdec.mat'
```

```
                elseif attr == 4 %Attribute 4 determined using utility
 equivalence (PE method)
                    attribute_utilities_m=
 [attribute_utilities_m(:,1),...

 attribute_utilities_m(:,3),attribute_utilities_m(:,2),attribute_utilities_m(:,4)]
                    attribute_utilities_t =
 array2table(attribute_utilities_m,...
                        'VariableNames',
{'xVal' 'u_x_down' 'u_x_up' 'DPrecision'});
                    disp(attribute_utilities_t);
                    % Fit optimal parameters for data considering
 uncertainty on u, all are descending
                    %Define contants and re-arrange data
                    disp('Optimising attribute function parameters and
 plotting ...');
                    x_v = attribute_utilities_m(2:(end-1),1)';
                    x_v = sort(x_v); %#ok<TRSRT>
                    u_m = attribute_utilities_m(2:(end-1),2:3)';
                    u_m = sort(u_m,2,'descend');
                    abeg = 0.5;
                    x0beg = 2;
                    xd=attribute_utilities_m(end,1); %Lowest value
                    xu=attribute_utilities_m(1,1); %Highest value
                    flagparam = [1 1];
                    flagfile = 1;
                    flagplot = 1;
                    %Fit parameters
                    [aopt,x0opt,HIopt,~,~,~]=optparam_udec...

 (abeg,x0beg,flagparam,x_v,u_m,xd,xu,flagplot,flagfile);
                    disp(strcat('Optimal attribute function parameters
 are: ...',num2str(aopt),...
                    '...and...',num2str(x0opt),'... with a fitting
 error of ...',num2str(HIopt)));
                    % Save fitting results for attribute from refresh
 of
                    % parameters
                    inp_filename2 = 'optpar_udec';

 load(inp_filename2,'aopt','x0opt','HIopt','flagparam');
                    out_filename2 = ['utils' '\','pump_attr_funct_'
 num2str(attr) '_params'];

 save(out_filename2,'aopt','x0opt','xd','xu','HIopt','flagparam');
                    disp(strcat('Attribute function parameters saved
 as: ... ',out_filename2));
                    delete 'optpar_udec.mat'
                elseif attr == 6
                    attribute_utilities_t =
 array2table(attribute_utilities_m,...
                        'VariableNames',data_names_c);
                    disp(attribute_utilities_t);
```

```
                    disp('A utility function is not fitted for this
attribute as it is binary.');
                end
                view_again = input('View another attribute? Y/N as
character');
                close;
            end
        elseif (choice == 2) %%View and modify values
            % View data as per previous choice
            inp_filename1_str=['utils' '\' 'survey_utils'];

load(inp_filename1_str,'attribute_names_c','data_names_c',...
                'utilities_matrix_m','ki_weights_m');
            modify_data = 'Y';
            while (modify_data == 'Y') || (modify_data == 'y') %Modify
data
                disp(strcat('Data loaded into workspace. Dataset size
is ...',...
                    num2str(size(utilities_matrix_m(:,2:end),1)),'...
rows by ...',...
                    num2str(size(utilities_matrix_m(:,2:end),2)),'...
columns.'));
                modify_indices_m = input(...
                    'Please input data rows and columns
to modify as well as new values in a matrix of the form
[row1,col1,newval1;row2,col2,newval2...].');
                disp(modify_indices_m);
                disp('Modifying values then pausing execution. Press
any key to continue.');
                [~,new_dataset_m] =
replace_values(utilities_matrix_m(:,2:end),...

modify_indices_m(:,1),modify_indices_m(:,2),modify_indices_m(:,3));
                disp('Old dataset:');
                disp(utilities_matrix_m(:,2:end));
                disp('New dataset with replacements:');
                disp(new_dataset_m);
                pause;
                %Prompt to save changes or discard
                update_data = input('Save changes? Y/N as character');
                if (update_data == 'Y') || (update_data == 'y')
                    utilities_matrix_m(:,2:end)=new_dataset_m;
                    out_filename1_str=['utils' '\' 'survey_utils'];

save(out_filename1_str,'attribute_names_c','data_names_c',...
                        'utilities_matrix_m','ki_weights_m');
                    input('Changes saved. Press any key to
continue.');
                elseif (update_data == 'N') || (update_data == 'n')
                    input('Changes NOT saved. Press any key to
continue.');
                end
                modify_data = input('Modify another range? Y/N as
character');
```

```
            end
        elseif (choice == 3) %Append or reduce dataset
            resize_dataset = 'Y';
            view_again = 'Y';
            inp_filename1_str=['utils' '\' 'survey_utils'];

load(inp_filename1_str,'attribute_names_c','data_names_c',...
                'utilities_matrix_m','ki_weights_m');
            while (view_again == 'Y') || (view_again == 'y')
                disp(strcat('Data loaded into workspace. Dataset size
is ...',...
                    num2str(size(utilities_matrix_m(:,2:end),1)),'...
rows by ...',...
                    num2str(size(utilities_matrix_m(:,2:end),2)),'...
columns.'));
                disp(strcat('This dataset is made up of utilities
for ...',...
                    num2str(length(attribute_names_c)),'...
attributes'));
                attr = menu('Please select an attribute to view
associated data:',...
                    attribute_names_c);
                disp(strcat('Displaying utility ranges for
attribute: ...',...
                    attribute_names_c{attr}));
                subset_v = utilities_matrix_m(:,1)== attr;
                attribute_utilities_m =
utilities_matrix_m(subset_v,2:end);
                attribute_utilities_t =
array2table(attribute_utilities_m,...
                    'VariableNames',data_names_c);
                disp(attribute_utilities_t);
                view_again = input('View another attribute? Y/N as
character');
            end
            disp(strcat('Proceeding to change size of dataset for
attribute:',...
                    attribute_names_c{attr}));
            resized_dataset_m = attribute_utilities_m;
            %Define the subset of utilities which won't be changed
            unchanged_subset_v = utilities_matrix_m(:,1)~= attr;
            unchanged_dataset_m =
utilities_matrix_m(unchanged_subset_v,2:end);
            while (resize_dataset == 'Y') || (resize_dataset == 'y')
                num_rows = size(resized_dataset_m,1);
                num_columns = size(resized_dataset_m,2);
                disp(strcat('Data loaded into workspace. Dataset size
is ...',...
                    num2str(num_rows),'... rows by ...',...
                    num2str(num_columns),'... columns.'));
                resized_dataset_t = array2table(resized_dataset_m,...
                    'VariableNames',data_names_c);
                example_t = resized_dataset_t{1,:};
                disp(resized_dataset_t);
```

```
                action = menu('What do you want to do?','1 = Append
dataset','2 = Extract subset of dataset');
                if action == 1
                    disp(...
                        'Appending data by measurement to dataset. You
are allowed to modify the number of rows only.');
                    disp(strcat('You should enter the data in rows
with...',...
                        num2str(num_columns),'... like the example
below:'));
                    disp(example_t);
                    append_prompt_str = strcat(...
                        'Please enter a matrix of the desired number
of rows and with exactly ...',...
                        num2str(num_columns), '...columns in the form
[a,b,c...]');
                    append_measurements_m = input(append_prompt_str);
                    resized_dataset_results_m =
[resized_dataset_m,append_measurements_m];
                    new_rows = size(resized_dataset_results_m,1);
                elseif action == 2
                    disp('Extracting subset of dataset. You are
allowed to modify the number of rows only.');
                    extract_rows_v = input(...
                        'Please enter rows of dataset to extract as a
vector of the form [a,b,c...].');
                    extract_rows_v = sort(extract_rows_v);
                    resized_dataset_results_m =
resized_dataset_m(extract_rows_v,:);
                    new_rows = size(resized_dataset_results_m,1);
                end
                disp(strcat('The resulting dataset is a matrix
with ...',...
                    num2str(new_rows),'... rows
and ...',num2str(num_columns),':'));
                resized_dataset_results_t = array2table(...

resized_dataset_results_m,'VariableNames',data_names_c);
                disp(resized_dataset_results_t);
                disp('Press any key to continue.');
                pause;
                resize_dataset = input('Resize dataset further? Y/N as
character');
            end
            save_resize = 'N'; %#ok<NASGU>
            save_resize = input('Save changes and overwrite dataset?
Y/N as character');
            if (save_resize == 'Y') || (save_resize == 'y')

utilities_matrix_m=[unchanged_dataset_m;resized_dataset_results_m];
                out_filename2_str=['utils' '\' 'survey_utils'];

save(out_filename2_str,'attribute_names_c','data_names_c',...
                    'utilities_matrix_m','ki_weights_m');
```

```
                input('Resized data saved. Press any key to
 continue.');
            elseif (save_resize == 'N') || (save_resize == 'n')
                input('Changes NOT saved. Press any key to
 continue.');
            end
        elseif (choice == 4)  %View ki weights
            inp_filename1_str=['utils' '\' 'survey_utils'];

 load(inp_filename1_str,'attribute_names_c','data_names_c',...
                'utilities_matrix_m','ki_weights_m');
            disp('The survey data has the following attribute weight
 ranges:');
            ki_weights_t =
 array2table(ki_weights_m,'VariableNames',...
                {'ki_lower','ki_upper'},'RowNames',attribute_names_c);
            disp(ki_weights_t);
            input('Returning to Utilities Database Menu. Press any key
 to continue.');
        elseif (choice == 5)  %Modify ki weights values
            inp_filename1_str=['utils' '\' 'survey_utils'];

 load(inp_filename1_str,'attribute_names_c','data_names_c',...
                'utilities_matrix_m','ki_weights_m');
            disp('The survey data has the following attribute weight
 ranges:');
            ki_weights_t =
 array2table(ki_weights_m,'VariableNames',...
                {'ki_lower','ki_upper'},'RowNames',attribute_names_c);
            disp(ki_weights_t);
            modify_values = input('Modify attribute weight values? Y/N
 as character');
            if (modify_values == 'Y') || (modify_values == 'y')
                modify_values_m = input(...
                    'Please enter the row indices,column
 indices and the new values to be used according to the example:
\n[row1,col1,value1;row2,col2,newvalue2...]\n');
                [~,new_ki_weights_m] = replace_values(...

 ki_weights_m,modify_values_m(:,1),modify_values_m(:,2),modify_values_m(:,3));
                new_ki_weights_t = array2table(new_ki_weights_m,...
                    'VariableNames',
{'ki_lower','ki_upper'},'RowNames',attribute_names_c);
                disp(new_ki_weights_t);
                save_changes = input('Save changes to attribute
 weights? Y/N as character');
                if (save_changes == 'Y') || (save_changes == 'y')
                    ki_weights_m = new_ki_weights_m;
                    out_filename1_str=['utils' '\' 'survey_utils'];
                    save(out_filename1_str,'attribute_names_c',...

  'data_names_c','utilities_matrix_m','ki_weights_m');
                    input('Changes saved.Press any key to continue.');
                elseif (save_changes ~= 'Y') || (save_changes ~= 'y')
```

```
                    input('Changes NOT saved. Press any key to
continue.');
                end
                modify_values = input(...
                    'Modify attribute weight values again? Y/N as
character'); %#ok<NASGU>
            end
            disp('Returning to Utilities Database Menu.');
        elseif (choice == 6) %View constants for pump
            %Load constants for calculation
            inp_filename1_str = ['utils' '\' 'constants_pump'];

load(inp_filename1_str,'constants_m','constants_names_c',...
                'cost_pump','cost_ship','eng_rate',...

 'insp_time','min_time','num_injured','sched_time','stop_time');
            inp_filename2_str = ['DATA1' '\' 'learning_data'];
            load(inp_filename2_str,'class_names_c');
            state_constants_t =
array2table(constants_m,'VariableNames',...
                constants_names_c,'RowNames',class_names_c);
            disp('Displaying state-specific constants for pump utility
calculations:');
            disp(state_constants_t);
            disp('Displaying individual constants:');
            disp(strcat(...
                'The cost of a new replacement pump in $AUD
is: ...',num2str(cost_pump)));
            disp(strcat(...
                'The cost of a new replacement vessel in $AUD
is: ...',num2str(cost_ship)));
            disp(strcat(...
                'The rate per hour for the person inspecting the pump
in $AUD is: ...',...
                num2str(eng_rate)));
            disp(strcat(...
                'The assumed time taken to inspect the pump in hours
is: ...',...
                num2str(insp_time)));
            disp(strcat(...
                'The time penalty for long maintenance tasks in hours
is: ...',...
                num2str(min_time)));
            disp(strcat(...
                'The assumed maximum number of people injured in the
event of a failure is: ...',...
                num2str(num_injured)));
            disp(strcat(...
                'The assumed time taken to schedule future maintenance
in hours is: ...',...
                num2str(sched_time)));
            disp(strcat(...
                'The assumed time taken to stop the pump in hours
is: ...',...
```

```
                num2str(stop_time)));
            input('Returning to Utilities Database Menu. Press any
key.');
        elseif (choice == 7) %Modify constants for pump
            %Load constants for calculation
            inp_filename1_str = ['utils' '\' 'constants_pump'];

load(inp_filename1_str,'constants_m','constants_names_c',...
                'cost_pump','cost_ship','eng_rate',...

 'insp_time','min_time','num_injured','sched_time','stop_time');
            inp_filename2_str = ['DATA1' '\' 'learning_data'];
            load(inp_filename2_str,'class_names_c');
            state_constants_t =
array2table(constants_m,'VariableNames',...
                constants_names_c,'RowNames',class_names_c);
            disp('Displaying state-specific constants for pump utility
calculations:');
            disp(state_constants_t);
            disp('Displaying individual constants:');
            disp(strcat('The cost of a new replacement pump in $AUD
is: ...',...
                num2str(cost_pump)));
            disp(strcat('The cost of a new replacement vessel in $AUD
is: ...',...
                num2str(cost_ship)));
            disp(strcat('The rate per hour for the person inspecting
the pump in $AUD is: ...',...
                num2str(eng_rate)));
            disp(strcat('The assumed time taken to inspect the pump in
hours is: ...',...
                num2str(insp_time)));
            disp(strcat('The time penalty for long maintenance tasks
in hours is: ...',...
                num2str(min_time)));
            disp(strcat('The assumed maximum number of people injured
in the event of a failure is: ...',...
                num2str(num_injured)));
            disp(strcat('The assumed time taken to schedule future
maintenance in hours is: ...',...
                num2str(sched_time)));
            disp(strcat('The assumed time taken to stop the pump in
hours is: ...',...
                num2str(stop_time)));
            modify_values = menu('Modify state-based constants or
individual constants?',...
                '1 = State-based Constants','2 = Individual
Constants','3 = Return to Utilities Database Menu');
            if modify_values == 1
                replace_state_constants = 'Y';
                while (replace_state_constants == 'Y') ||
(replace_state_constants == 'y')
                    modify_values_m = input(...
```

```
                        'Please enter the row indices,column
 indices and the new values to be used according to the example:
\n[row1,col1,value1;row2,col2,newvalue2...]\n');
                    [~,new_constants_m] =
 replace_values(constants_m,...

 modify_values_m(:,1),modify_values_m(:,2),modify_values_m(:,3));
                    new_state_constants_t =
 array2table(new_constants_m,...

  'VariableNames',constants_names_c,'RowNames',class_names_c);
                    disp('Displaying modified-state-specific constants
 for pump utility calculations:');
                    disp(new_state_constants_t);
                    save_mods_state = input('Save changes to state-
based constants? Y/N as character');
                    replace_state_constants = input('Modify another
 value? Y/N as character');
                    if (save_mods_state == 'Y') || (save_mods_state
 == 'y')
                        constants_m = new_constants_m;
                        out_filename2_str =
 ['utils' '\' 'constants_pump'];
                        save(out_filename2_str,'constants_m',...

  'constants_names_c','cost_pump','cost_ship','eng_rate',...

  'insp_time','min_time','num_injured','sched_time','stop_time');
                        disp('Changes saved');
                    end
                end
                disp('Changes NOT saved.');
                disp('Returning to Utilities Database Menu.');
            elseif modify_values == 2
                replace_individual = 'Y';
                while (replace_individual == 'Y') ||
 (replace_individual == 'y')
                    modify_menu_options_c =
 {'cost_pump','cost_ship','eng_rate',...

  'insp_time','min_time','num_injured','sched_time','stop_time'};
                    individual_constants_v =
 [cost_pump,cost_ship,eng_rate,...

 insp_time,min_time,num_injured,sched_time,stop_time];
                    mod_ind = menu('Please select the value to
 modify:',modify_menu_options_c);
                    disp(strcat('The current value
 is: ...',num2str(individual_constants_v(mod_ind))));
                    replace_val = input('Please enter a new value for
 the constant:\n');
                    individual_constants_v(mod_ind) = replace_val;
                    disp('Value overwritten.');
```

```
                    save_mods_ind = input('Save current changes to
individual constants? Y/N as character');
                    if (save_mods_ind == 'Y') || (save_mods_ind
== 'y')
                        constants_m = new_constants_m;
                        cost_pump=individual_constants_v(1);
                        cost_ship=individual_constants_v(2);
                        eng_rate=individual_constants_v(3);
                        insp_time=individual_constants_v(4);
                        min_time=individual_constants_v(5);
                        num_injured=individual_constants_v(6);
                        sched_time=individual_constants_v(7);
                        stop_time=individual_constants_v(8);
                        out_filename2_str =
['utils' '\' 'constants_pump'];
                        save(out_filename2_str,'constants_m',...

 'constants_names_c','cost_pump','cost_ship','eng_rate',...

 'insp_time','min_time','num_injured','sched_time','stop_time');
                        disp('Changes saved');
                    end
                    replace_individual=input('Modify another
individual value? Y/N as character\n');
                end
                disp('Changes NOT saved.');
                disp('Returning to Utilities Database Menu.');
            elseif modify_values == 3
                disp('Returning to Utilities Database Menu.');
            end
            %%%%

        elseif (choice == 8)
            inp_filename3_str=['utils' '\' 'survey_utils_default'];

load(inp_filename3_str,'attribute_names_c','data_names_c',...
                'utilities_matrix_m','ki_weights_m');
            out_filename3_str=['utils' '\' 'survey_utils'];

save(out_filename3_str,'attribute_names_c','data_names_c',...
                'utilities_matrix_m','ki_weights_m');
            inp_filename4_str =
['utils' '\' 'constants_pump_default'];

load(inp_filename4_str,'constants_m','constants_names_c',...
                'cost_pump','cost_ship','eng_rate',...

 'insp_time','min_time','num_injured','sched_time','stop_time');
            out_filename4_str = ['utils' '\' 'constants_pump'];

save(out_filename4_str,'constants_m','constants_names_c',...
                'cost_pump','cost_ship','eng_rate',...

 'insp_time','min_time','num_injured','sched_time','stop_time');
```

```
            disp('Defaults restored. Returning to Utilities Database
Menu.');
        elseif (choice == 9) %View only (also calculates optimal data
fitted parameters)
            view_again = 'Y';
            inp_filename1_str=['utils' '\' 'test_utils'];

load(inp_filename1_str,'attribute_names_c','data_names_c',...
                'utilities_matrix_m','ki_weights_m');
            while (view_again == 'Y') || (view_again == 'y')
                disp(strcat('Data loaded into workspace. Dataset size
is ...',...
                    num2str(size(utilities_matrix_m(:,2:end),1)),'...
rows by ...',...
                    num2str(size(utilities_matrix_m(:,2:end),2)),'...
columns.'));
                disp(strcat('This dataset is made up of utilities
for ...',...
                    num2str(length(attribute_names_c)),'...
attributes'));
                attr = menu('Please select an attribute to view
associated data:',...
                    attribute_names_c);
                disp(strcat('Displaying utility ranges for
attribute: ...',...
                    attribute_names_c{attr}));
                subset_v = utilities_matrix_m(:,1)== attr;
                attribute_utilities_m =
utilities_matrix_m(subset_v,2:end);
                if (attr ~=4) && (attr ~=6)
                    attribute_utilities_t = array2table(...

attribute_utilities_m,'VariableNames',data_names_c);
                    disp(attribute_utilities_t);
                    % Fit optimal parameters for data considering
uncertainty on x, all are descending
                    %Define contants and re-arrange data
                    disp('Optimising attribute function parameters and
plotting ...');
                    x_m = attribute_utilities_m(2:(end-1),2:3)';
                    x_m = sort(x_m,2);
                    u_v = attribute_utilities_m(2:(end-1),end)';
                    u_v = sort(u_v,'descend');
                    abeg = 0.5;
                    x0beg = 2;
                    xd=attribute_utilities_m(end,2); %Lowest value
                    xu=attribute_utilities_m(1,3); %Highest value
                    flagparam = [1 1];
                    flagfile = 1;
                    flagplot = 1;
                    %Fit parameters
                    [aopt,x0opt,HIopt,~,~,~]=optparam_xdec...

(abeg,x0beg,flagparam,x_m,u_v,xd,xu,flagplot,flagfile);
```

```
                    disp(strcat('Optimal attribute function parameters
are: ...',...
                        num2str(aopt),'...and...',num2str(x0opt),...
                        '... with a fitting error
of ...',num2str(HIopt)));
                    % Save fitting results for attribute from refresh
of
                    % parameters
                    inp_filename2 = 'optpar_xdec';

load(inp_filename2,'aopt','x0opt','HIopt','flagparam');
                    out_filename2 =
['utils' '\' 'test_attr_funct_' ...
                        num2str(attr) '_params'];
                    save(out_filename2,'aopt','x0opt','xd','xu',...
                        'HIopt','flagparam');
                    disp(strcat('Attribute function parameters saved
as: ... ',out_filename2));
                    delete 'optpar_xdec.mat'
                elseif attr == 4 %Attribute 4 determined using utility
equivalence (PE method)
                    attribute_utilities_m=
[attribute_utilities_m(:,1),...

attribute_utilities_m(:,3),attribute_utilities_m(:,2),attribute_utilities_m(:,4)]
                    attribute_utilities_t = array2table(...
                        attribute_utilities_m,'VariableNames',
{'xVal' 'u_x_down' 'u_x_up' 'DPrecision'});
                    disp(attribute_utilities_t);
                    % Fit optimal parameters for data considering
uncertainty on u, all are descending
                    %Define contants and re-arrange data
                    disp('Optimising attribute function parameters and
plotting ...');
                    x_v = attribute_utilities_m(2:(end-1),1)';
                    x_v = sort(x_v); %#ok<TRSRT>
                    u_m = attribute_utilities_m(2:(end-1),2:3)';
                    u_m = sort(u_m,2,'descend');
                    abeg = 0.5;
                    x0beg = 2;
                    xd=attribute_utilities_m(end,1); %Lowest value
                    xu=attribute_utilities_m(1,1); %Highest value
                    flagparam = [1 1];
                    flagfile = 1;
                    flagplot = 1;
                    %Fit parameters
                    [aopt,x0opt,HIopt,~,~,~]=optparam_udec...

(abeg,x0beg,flagparam,x_v,u_m,xd,xu,flagplot,flagfile);
                    disp(strcat('Optimal attribute function parameters
are: ...',...
                        num2str(aopt),'...and...',num2str(x0opt),...
                        '... with a fitting error
of ...',num2str(HIopt)));
```

```
                    % Save fitting results for attribute from refresh
of
                    % parameters
                    inp_filename2 = 'optpar_udec';

load(inp_filename2,'aopt','x0opt','HIopt','flagparam');
                    out_filename2 = ['utils' '\' 'test_attr_funct_'
num2str(attr) '_params'];

save(out_filename2,'aopt','x0opt','xd','xu','HIopt','flagparam');
                    disp(strcat('Attribute function parameters saved
as: ... ',out_filename2));
                    delete 'optpar_udec.mat'
                elseif attr == 6
                    attribute_utilities_t = array2table(...

attribute_utilities_m,'VariableNames',data_names_c);
                    disp(attribute_utilities_t);
                    disp('A utility function is not fitted for this
attribute as it is binary.');
                end
                view_again = input('View another attribute? Y/N as
character');
                close;
            end
        elseif (choice == 10) %View and modify values
            % View data as per previous choice
            inp_filename1_str=['utils' '\' 'test_utils'];

load(inp_filename1_str,'attribute_names_c','data_names_c',...
                'utilities_matrix_m','ki_weights_m');
            modify_data = 'Y';
            while (modify_data == 'Y') || (modify_data == 'y') %Modify
data
                disp(strcat('Data loaded into workspace. Dataset size
is ...',...
                    num2str(size(utilities_matrix_m(:,2:end),1)),'...
rows by ...',...
                    num2str(size(utilities_matrix_m(:,2:end),2)),'...
columns.'));
                modify_indices_m = input(...
                    'Please input data rows and columns
to modify as well as new values in a matrix of the form
[row1,col1,newval1;row2,col2,newval2...].');
                disp(modify_indices_m);
                disp('Modifying values then pausing execution. Press
any key to continue.');
                [~,new_dataset_m] = replace_values(...

utilities_matrix_m(:,2:end),modify_indices_m(:,1),...
                    modify_indices_m(:,2),modify_indices_m(:,3));
                disp('Old dataset:');
                disp(utilities_matrix_m(:,2:end));
                disp('New dataset with replacements:');
```

```
                disp(new_dataset_m);
                pause;
                %Prompt to save changes or discard
                update_data = input('Save changes? Y/N as character');
                if (update_data == 'Y') || (update_data == 'y')
                    utilities_matrix_m(:,2:end)=new_dataset_m;
                    out_filename1_str=['utils' '\' 'test_utils'];
                    save(out_filename1_str,'attribute_names_c',...
 'data_names_c','utilities_matrix_m','ki_weights_m');
                    input('Changes saved. Press any key to
continue.');
                elseif (update_data == 'N') || (update_data == 'n')
                    input('Changes NOT saved. Press any key to
continue.');
                end
                modify_data = input('Modify another range? Y/N as
character');
            end
        elseif (choice == 11) %Append or reduce dataset
            resize_dataset = 'Y';
            view_again = 'Y';
            inp_filename1_str=['utils' '\' 'test_utils'];

load(inp_filename1_str,'attribute_names_c','data_names_c',...
                'utilities_matrix_m','ki_weights_m');
            while (view_again == 'Y') || (view_again == 'y')
                disp(strcat('Data loaded into workspace. Dataset size
is ...',...
                    num2str(size(utilities_matrix_m(:,2:end),1)),'...
rows by ...',...
                    num2str(size(utilities_matrix_m(:,2:end),2)),'...
columns.'));
                disp(strcat('This dataset is made up of utilities
for ...',...
                    num2str(length(attribute_names_c)),'...
attributes'));
                attr = menu('Please select an attribute to view
associated data:',...
                    attribute_names_c);
                disp(strcat('Displaying utility ranges for
attribute: ...',...
                    attribute_names_c{attr}));
                subset_v = utilities_matrix_m(:,1)== attr;
                attribute_utilities_m =
utilities_matrix_m(subset_v,2:end);
                attribute_utilities_t =
array2table(attribute_utilities_m,...
                    'VariableNames',data_names_c);
                disp(attribute_utilities_t);
                view_again = input('View another attribute? Y/N as
character');
            end
```

```
            disp(strcat('Proceeding to change size of dataset for
attribute:',...
                attribute_names_c{attr}));
            resized_dataset_m = attribute_utilities_m;
            %Define the subset of utilities which won't be changed
            unchanged_subset_v = utilities_matrix_m(:,1)~= attr;
            unchanged_dataset_m =
utilities_matrix_m(unchanged_subset_v,2:end);
            while (resize_dataset == 'Y') || (resize_dataset == 'y')
                num_rows = size(resized_dataset_m,1);
                num_columns = size(resized_dataset_m,2);
                disp(strcat('Data loaded into workspace. Dataset size
is ...',...
                    num2str(num_rows),'... rows by ...',...
                    num2str(num_columns),'... columns.'));
                resized_dataset_t = array2table(resized_dataset_m,...
                    'VariableNames',data_names_c);
                example_t = resized_dataset_t{1,:};
                disp(resized_dataset_t);
                action = menu('What do you want to do?',...
                    '1 = Append dataset','2 = Extract subset of
dataset');
                if action == 1
                    disp(...
                        'Appending data by measurement to dataset. You
are allowed to modify the number of rows only.');
                    disp(strcat('You should enter the data in rows
with...',...
                        num2str(num_columns),'... like the example
below:'));
                    disp(example_t);
                    append_prompt_str = strcat(...
                        'Please enter a matrix of the desired number
of rows and with exactly ...',...
                        num2str(num_columns), '...columns in the form
[a,b,c...]');
                    append_measurements_m = input(append_prompt_str);
                    resized_dataset_results_m =
[resized_dataset_m,append_measurements_m];
                    new_rows = size(resized_dataset_results_m,1);
                elseif action == 2
                    disp('Extracting subset of dataset. You are
allowed to modify the number of rows only.');
                    extract_rows_v = input(...
                        'Please enter rows of dataset to extract as a
vector of the form [a,b,c...].');
                    extract_rows_v = sort(extract_rows_v);
                    resized_dataset_results_m =
resized_dataset_m(extract_rows_v,:);
                    new_rows = size(resized_dataset_results_m,1);
                end
                disp(strcat('The resulting dataset is a matrix
with ...',...
```

```
                    num2str(new_rows),'... rows
and ...',num2str(num_columns),':'));
                resized_dataset_results_t =
array2table(resized_dataset_results_m,...
                    'VariableNames',data_names_c);
                disp(resized_dataset_results_t);
                disp('Press any key to continue.');
                pause;
                resize_dataset = input('Resize dataset further? Y/N as
character');
            end
            save_resize = 'N'; %#ok<NASGU>
            save_resize = input('Save changes and overwrite dataset?
Y/N as character');
            if (save_resize == 'Y') || (save_resize == 'y')

utilities_matrix_m=[unchanged_dataset_m;resized_dataset_results_m];
                out_filename2_str=['utils' '\' 'test_utils'];

save(out_filename2_str,'attribute_names_c','data_names_c',...
                    'utilities_matrix_m','ki_weights_m');
                input('Resized data saved. Press any key to
continue.');
            elseif (save_resize == 'N') || (save_resize == 'n')
                input('Changes NOT saved. Press any key to
continue.');
            end
        elseif (choice == 12) %View Test ki Weights
            inp_filename1_str=['utils' '\' 'test_utils'];

load(inp_filename1_str,'attribute_names_c','data_names_c',...
                'utilities_matrix_m','ki_weights_m');
            disp('The survey data has the following attribute weight
ranges:');
            ki_weights_t =
array2table(ki_weights_m,'VariableNames',...
                {'ki_lower','ki_upper'},'RowNames',attribute_names_c);
            disp(ki_weights_t);
            input('Returning to Utilities Database Menu. Press any key
to continue.');
        elseif (choice == 13) %Modify Test ki Weights
            inp_filename1_str=['utils' '\' 'test_utils'];

load(inp_filename1_str,'attribute_names_c','data_names_c',...
                'utilities_matrix_m','ki_weights_m');
            disp('The survey data has the following attribute weight
ranges:');
            ki_weights_t =
array2table(ki_weights_m,'VariableNames',...
                {'ki_lower','ki_upper'},'RowNames',attribute_names_c);
            disp(ki_weights_t);
            modify_values = input('Modify attribute weight values? Y/N
as character');
            if (modify_values == 'Y') || (modify_values == 'y')
```

```
                modify_values_m = input(...
                    'Please enter the row indices,column
 indices and the new values to be used according to the example:
\n[row1,col1,value1;row2,col2,newvalue2...]\n');
                [~,new_ki_weights_m] = replace_values(ki_weights_m,...

 modify_values_m(:,1),modify_values_m(:,2),modify_values_m(:,3));
                new_ki_weights_t = array2table(new_ki_weights_m,...
                    'VariableNames',
{'ki_lower','ki_upper'},'RowNames',attribute_names_c);
                disp(new_ki_weights_t);
                save_changes = input('Save changes to attribute
 weights? Y/N as character');
                if (save_changes == 'Y') || (save_changes == 'y')
                    ki_weights_m = new_ki_weights_m;
                    out_filename1_str=['utils' '\' 'test_utils'];
                    save(out_filename1_str,'attribute_names_c',...

  'data_names_c','utilities_matrix_m','ki_weights_m');
                    input('Changes saved.Press any key to continue.');
                elseif (save_changes ~= 'Y') || (save_changes ~= 'y')
                    input('Changes NOT saved. Press any key to
 continue.');
                end
                modify_values = input(...
                    'Modify attribute weight values again? Y/N as
 character'); %#ok<NASGU>
            end
            disp('Returning to Utilities Database Menu.');
        elseif (choice == 14) %View constants for test data
            inp_filename1_str = ['utils' '\' 'constants_test'];

 load(inp_filename1_str,'constants_m','constants_names_c',...
                'cost_pump','cost_ship','eng_rate',...

  'insp_time','min_time','num_injured','sched_time','stop_time');
            inp_filename2_str = ['DATA1' '\' 'learning_data'];
            load(inp_filename2_str,'class_names_c');
            state_constants_t =
 array2table(constants_m,'VariableNames',...
                constants_names_c,'RowNames',class_names_c);
            disp('Displaying state-specific constants for pump utility
 calculations:');
            disp(state_constants_t);
            disp('Displaying individual constants:');
            disp(strcat(...
                'The cost of a new replacement pump in $AUD
 is: ...',num2str(cost_pump)));
            disp(strcat(...
                'The cost of a new replacement vessel in $AUD
 is: ...',num2str(cost_ship)));
            disp(strcat(...
                'The rate per hour for the person inspecting the pump
 in $AUD is: ...',...
```

```
                num2str(eng_rate)));
            disp(strcat(...
                'The assumed time taken to inspect the pump in hours
is: ...',...
                num2str(insp_time)));
            disp(strcat(...
                'The time penalty for long maintenance tasks in hours
is: ...',...
                num2str(min_time)));
            disp(strcat(...
                'The assumed maximum number of people injured in the
event of a failure is: ...',...
                num2str(num_injured)));
            disp(strcat(...
                'The assumed time taken to schedule future maintenance
in hours is: ...',...
                num2str(sched_time)));
            disp(strcat(...
                'The assumed time taken to stop the pump in hours
is: ...',...
                num2str(stop_time)));
            input('Returning to Utilities Database Menu. Press any
key.');
        elseif (choice == 15) %Modify constants for test data
            inp_filename1_str = ['utils' '\' 'constants_test'];

load(inp_filename1_str,'constants_m','constants_names_c',...
                'cost_pump','cost_ship','eng_rate',...

 'insp_time','min_time','num_injured','sched_time','stop_time');
            inp_filename2_str = ['DATA1' '\' 'learning_data'];
            load(inp_filename2_str,'class_names_c');
            state_constants_t =
array2table(constants_m,'VariableNames',...
                constants_names_c,'RowNames',class_names_c);
            disp('Displaying state-specific constants for pump utility
calculations:');
            disp(state_constants_t);
            disp('Displaying individual constants:');
            disp(strcat(...
                'The cost of a new replacement pump in $AUD
is: ...',...
                num2str(cost_pump)));
            disp(strcat(...
                'The cost of a new replacement vessel in $AUD
is: ...',...
                num2str(cost_ship)));
            disp(strcat(...
                'The rate per hour for the person inspecting the pump
in $AUD is: ...',...
                num2str(eng_rate)));
            disp(strcat(...
                'The assumed time taken to inspect the pump in hours
is: ...',...
```

```
                num2str(insp_time)));
            disp(strcat(...
                'The time penalty for long maintenance tasks in hours
 is: ...',...
                num2str(min_time)));
            disp(strcat(...
                'The assumed maximum number of people injured in the
 event of a failure is: ...',...
                num2str(num_injured)));
            disp(strcat(...
                'The assumed time taken to schedule future maintenance
 in hours is: ...',...
                num2str(sched_time)));
            disp(strcat(...
                'The assumed time taken to stop the pump in hours
 is: ...',...
                num2str(stop_time)));
            modify_values = menu('Modify state-based constants or
 individual constants?',...
                '1 = State-based Constants','2 = Individual
 Constants',...
                '3 = Return to Utilities Database Menu');
            if modify_values == 1
                replace_state_constants = 'Y';
                while (replace_state_constants == 'Y') ||
 (replace_state_constants == 'y')
                    modify_values_m = input(...
                        'Please enter the row indices,column
 indices and the new values to be used according to the example:
\n[row1,col1,value1;row2,col2,newvalue2...]\n');
                    [~,new_constants_m] =
 replace_values(constants_m,...

 modify_values_m(:,1),modify_values_m(:,2),modify_values_m(:,3));
                    new_state_constants_t =
 array2table(new_constants_m,...

  'VariableNames',constants_names_c,'RowNames',class_names_c);
                    disp('Displaying modified-state-specific constants
 for pump utility calculations:');
                    disp(new_state_constants_t);
                    save_mods_state = input('Save changes to state-
based constants? Y/N as character');
                    replace_state_constants = input('Modify another
 value? Y/N as character');
                    if (save_mods_state == 'Y') || (save_mods_state
 == 'y')
                        constants_m = new_constants_m;
                        out_filename2_str =
 ['utils' '\' 'constants_test'];
                        save(out_filename2_str,'constants_m',...

  'constants_names_c','cost_pump','cost_ship','eng_rate',...
```

```
 'insp_time','min_time','num_injured','sched_time','stop_time');
                        disp('Changes saved');
                    end
                end
                disp('Changes NOT saved.');
                disp('Returning to Utilities Database Menu.');
            elseif modify_values == 2
                replace_individual = 'Y';
                while (replace_individual == 'Y') ||
(replace_individual == 'y')
                    modify_menu_options_c =
{'cost_pump','cost_ship','eng_rate',...

 'insp_time','min_time','num_injured','sched_time','stop_time'};
                    individual_constants_v =
[cost_pump,cost_ship,eng_rate,...

insp_time,min_time,num_injured,sched_time,stop_time];
                    mod_ind = menu('Please select the value to
modify:',modify_menu_options_c);
                    disp(strcat('The current value
is: ...',num2str(individual_constants_v(mod_ind))));
                    replace_val = input('Please enter a new value for
the constant:\n');
                    individual_constants_v(mod_ind) = replace_val;
                    disp('Value overwritten.');
                    save_mods_ind = input('Save current changes to
individual constants? Y/N as character');
                    if (save_mods_ind == 'Y') || (save_mods_ind
== 'y')
                        constants_m = new_constants_m;
                        cost_pump=individual_constants_v(1);
                        cost_ship=individual_constants_v(2);
                        eng_rate=individual_constants_v(3);
                        insp_time=individual_constants_v(4);
                        min_time=individual_constants_v(5);
                        num_injured=individual_constants_v(6);
                        sched_time=individual_constants_v(7);
                        stop_time=individual_constants_v(8);
                        out_filename2_str =
['utils' '\' 'constants_test'];
                        save(out_filename2_str,'constants_m',...

 'constants_names_c','cost_pump','cost_ship','eng_rate',...

 'insp_time','min_time','num_injured','sched_time','stop_time');
                        disp('Changes saved');
                    end
                    replace_individual=input('Modify another
individual value? Y/N as character\n');
                end
                disp('Changes NOT saved.');
                disp('Returning to Utilities Database Menu.');
```

```
            elseif modify_values == 3
                disp('Returning to Utilities Database Menu.');
            end
        elseif (choice == 16)
            inp_filename3_str=['utils' '\' 'test_utils_default'];

 load(inp_filename3_str,'attribute_names_c','data_names_c','utilities_matrix_m','k
            out_filename3_str=['utils' '\' 'test_utils'];

 save(out_filename3_str,'attribute_names_c','data_names_c','utilities_matrix_m','k
            inp_filename4_str =
 ['utils' '\' 'constants_test_default'];

 load(inp_filename4_str,'constants_m','constants_names_c','cost_pump','cost_ship',

  'insp_time','min_time','num_injured','sched_time','stop_time');
            out_filename4_str = ['utils' '\' 'constants_test'];

 save(out_filename4_str,'constants_m','constants_names_c','cost_pump','cost_ship',

  'insp_time','min_time','num_injured','sched_time','stop_time');
            disp('Defaults restored. Returning to Utilities Database
 Menu.');
        end
    end
    input('Returning to Main Menu. Press any key to continue.');
end
```

Published with MATLAB® R2018b

APPENDIX D - Data Processing Procedure

The present Appendix presents all steps involved in processing measurement data into classification learning data or CM data as outlined in Section 3.7 previously.

Data processing was completed according to the following steps:

1. *Export all vibration data files from Commtest vb7 to Ascent 2015 software on a PC.*
2. *Save vibration FFT data files as .csv from all locations (Refer to Table 3-4 and Table 3-7).*
3. *Save the vibration waveform data file from the pump casing location only as .csv.*

This location is considered in the present study as it is used in another (Sakthivel et al., 2010).

4. *Split the vibration FFT and waveform data into individual measurement files.*

This was completed using a macro script in Excel.

5. *Rename all files according to the measurement location and the time they were taken, saving as .xlsx.*

This was also completed using a macro script in Excel.

6. *Using the shaft RPM measurement taken for the current experiment or CM measurement, process all FFT files using a MATLAB script to obtain Excel files which contain several characteristic FFT amplitudes.*

The Blade Pass Frequency (BPF), the fundamental frequency and multiple harmonics of the fundamental frequency (2x, 3x 4x, 5x, 6x, 7x, 8x, 12x, 20x, 36x and 37x) are calculated for the present Thesis as described in Section 3.7.1. Vibration FFT measurement processing results in 65 features from all five measurement locations.

7. *Process all velocity waveform data through a MATLAB script to obtain Excel files which contain descriptive statistics of the waveform (Kamiel, 2015; Sakthivel et al., 2010).*

Descriptive statistics include: mean, standard deviation, standard error, median, variance, skewness, kurtosis, range, minimum, maximum and sum and were used in the referenced studies to describe

the characteristics of pump faults. Non-dimensionless statistics in the present Thesis are obtained in units of mms^{-1}.

8. *Import casing and shaft temperature measurements from thermal imaging camera into FLIR processing software.*
9. *Save one point of temperature data per location (Refer to Table 3-5 and Table 3-9) each minute as an Excel .xlsx file.*
10. *Combine the processed vibration FFT files, processed vibration waveform files, temperature, pressure and remaining measurements into one spreadsheet, and save this as an Excel file.*

APPENDIX E - Examples of Processed Experimental and CM Vibration Data

The present Appendix E provides an example of some of the experimental and CM vibration data collected and processed as outlined previously in Chapter 3.

The following data represent the final sample collected from the 'pump casing' position for the given experiment or CM instance. Experimental data are presented according to their name and dataset number given previously in Chapter 4, Table 4-2. Grey data in the following Figures are the Dataset "1" class, 'Vessel Alongside, No Engines Running' which have been plotted for comparison.

All subsequent data processing steps are outlined in Chapter 3, Section 3.7 or Appendix D and processing software presented in Appendix C.

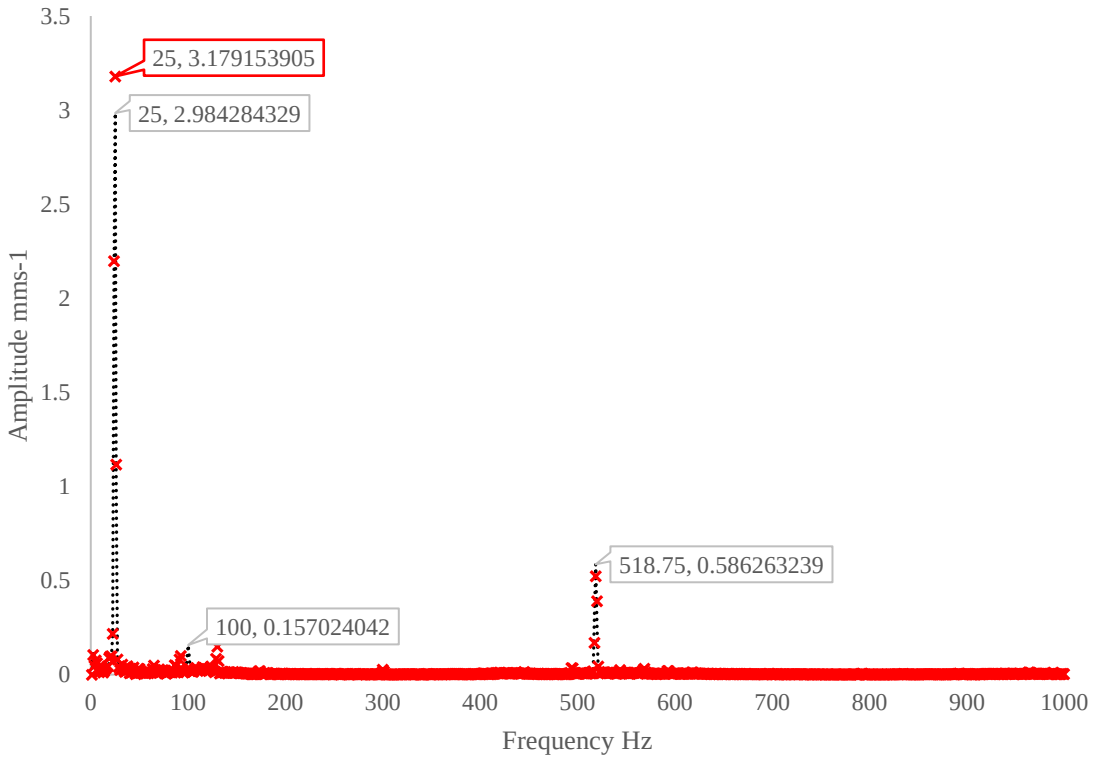
E.1 Velocity FFT Data

E.1.1 Experimental Data

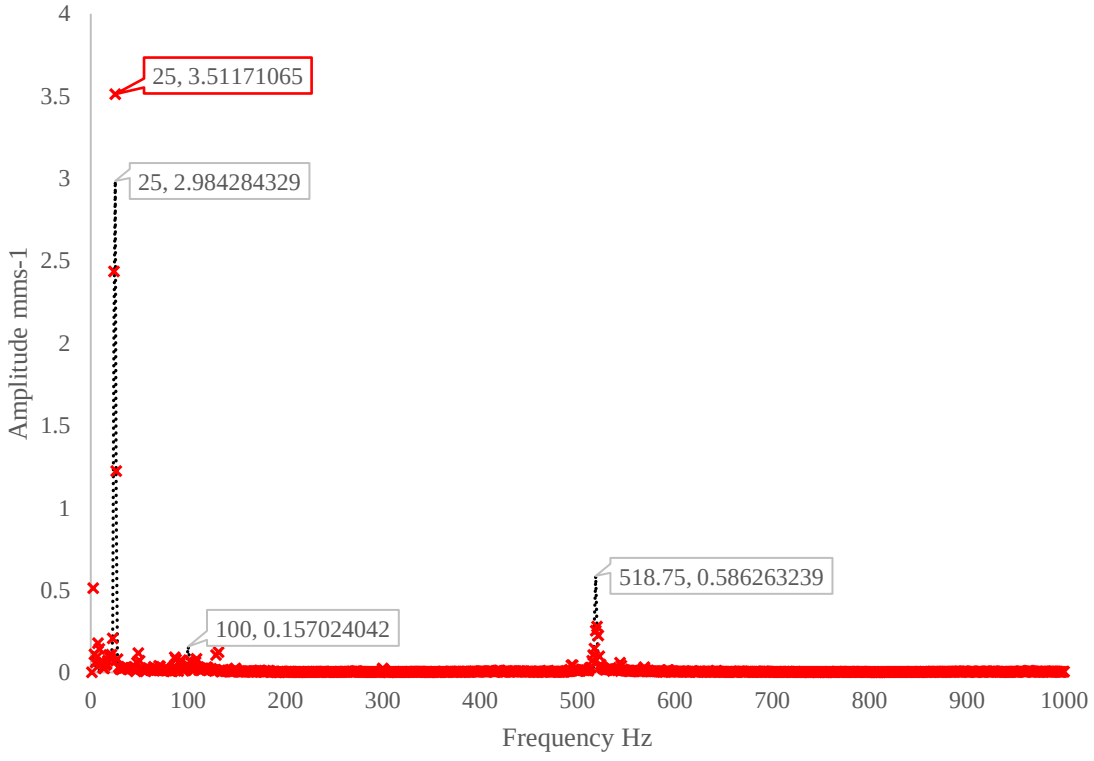
The Test pump configuration as described in Chapter 3, was used to obtain all experimental data displayed within this section.

In the following Figures E-1 to E-9, the calculated values for f_0 and BPF are 25 Hz and 520 Hz respectively.

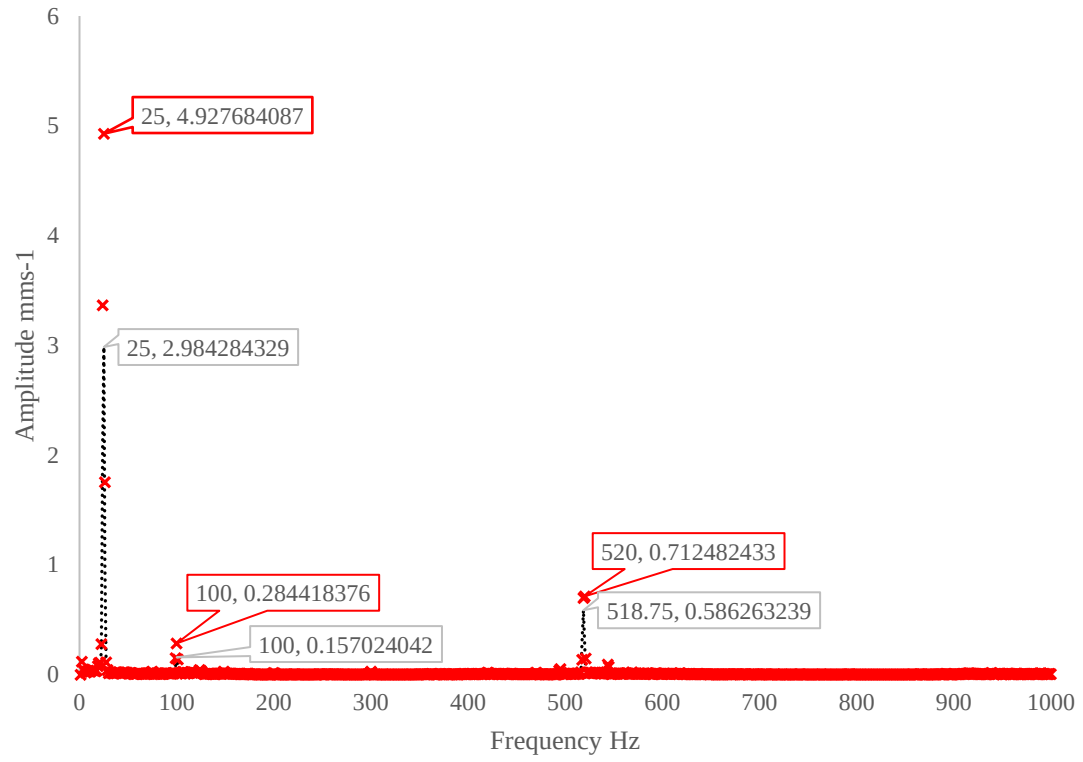
2	Figure E-1: Vessel Alongside, Engines Running - Velocity FFT



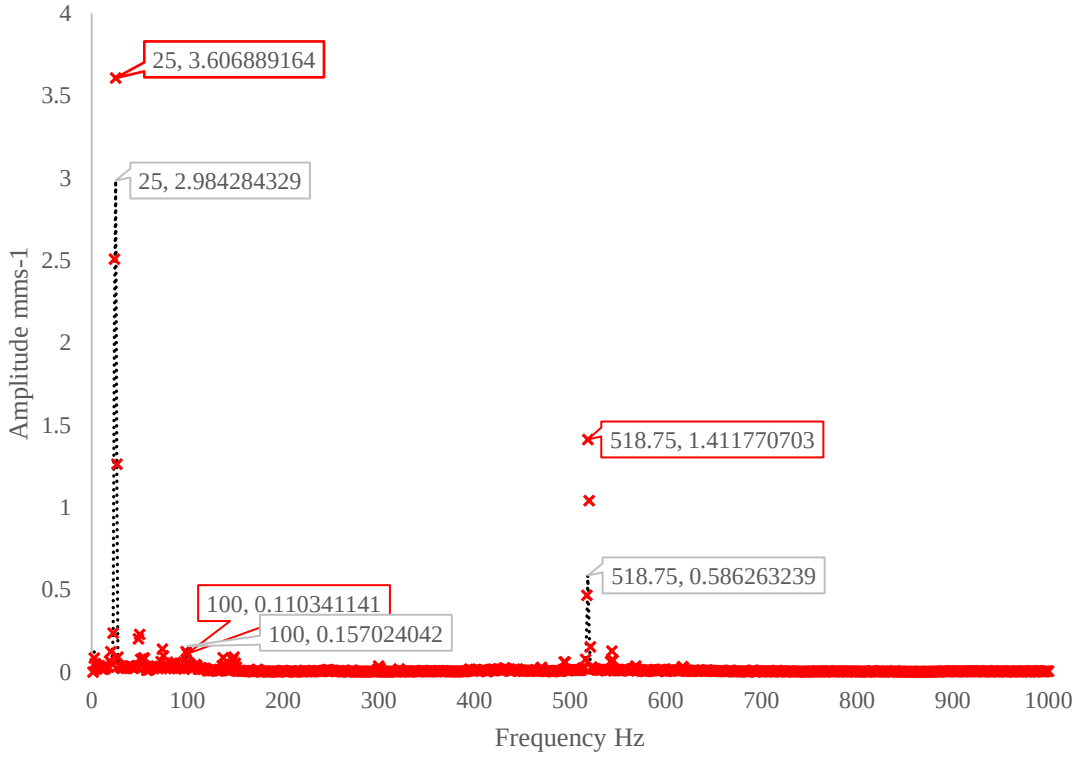
3	Figure E-2: Vessel at Sea - Velocity FFT



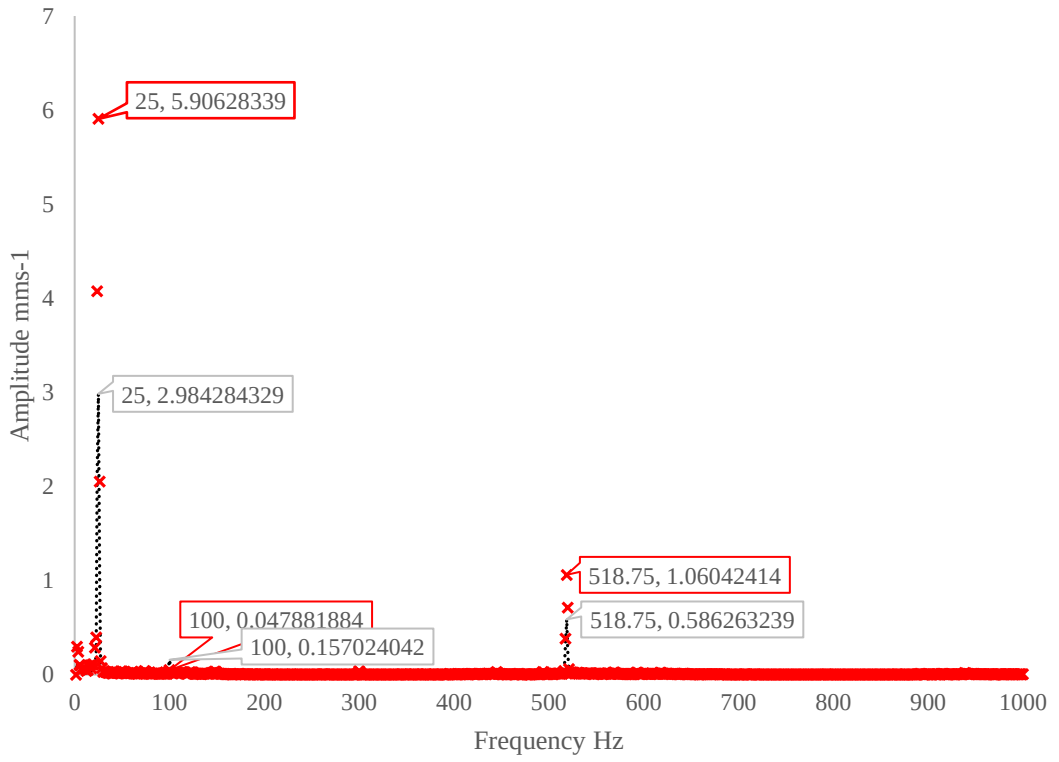
4	Figure E-3: Worn Impeller- Velocity FFT



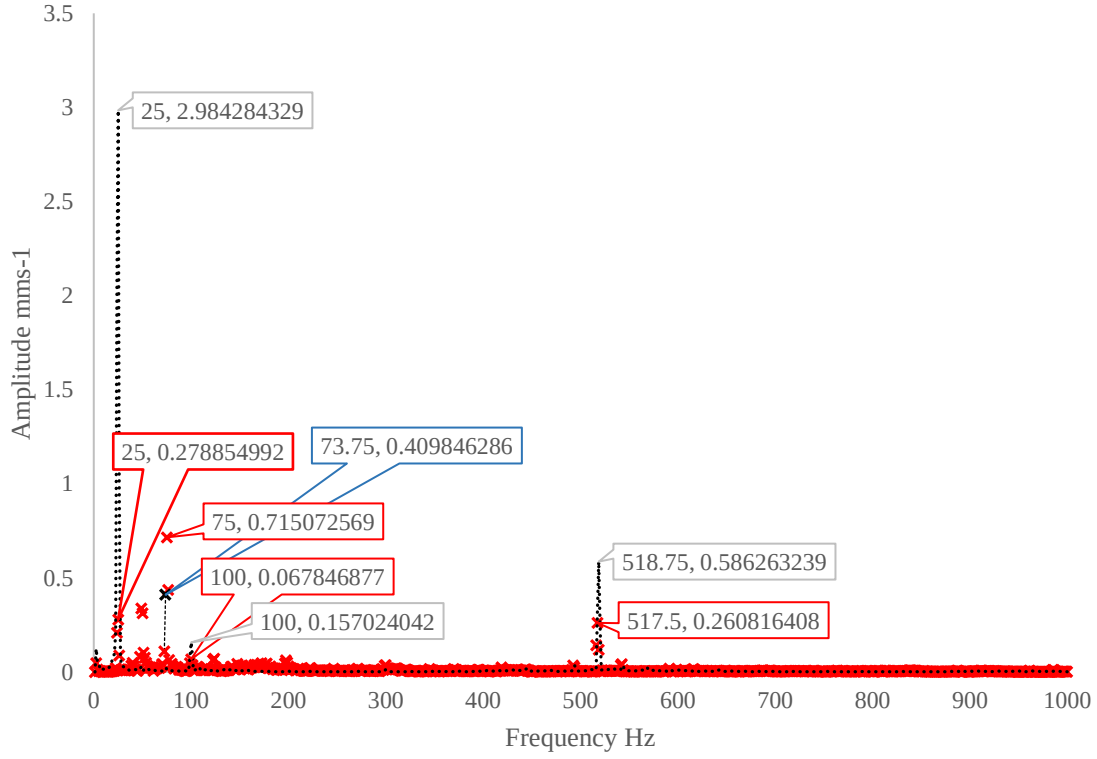
5	Figure E-4: Loose Packing- Velocity FFT



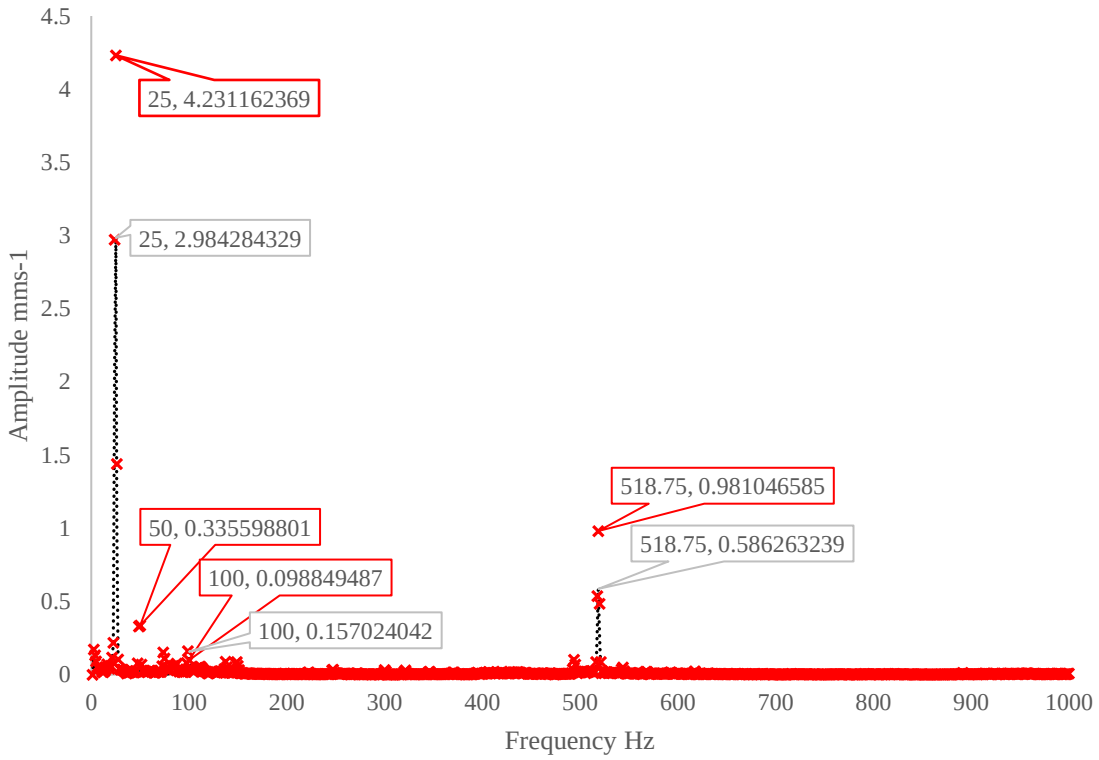
6	Figure E-5: Damaged Pump Drive-End Bearing- Velocity FFT



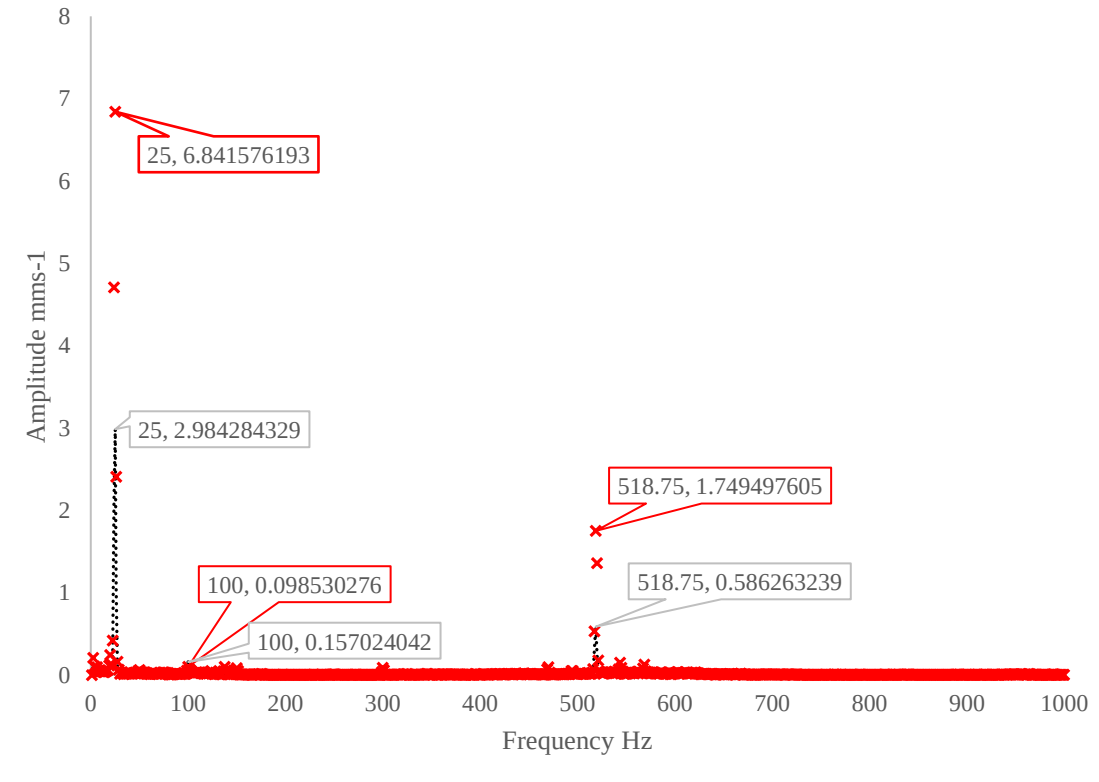
7	Figure E-6: Worn Pump Drive-End Bearing- Velocity FFT



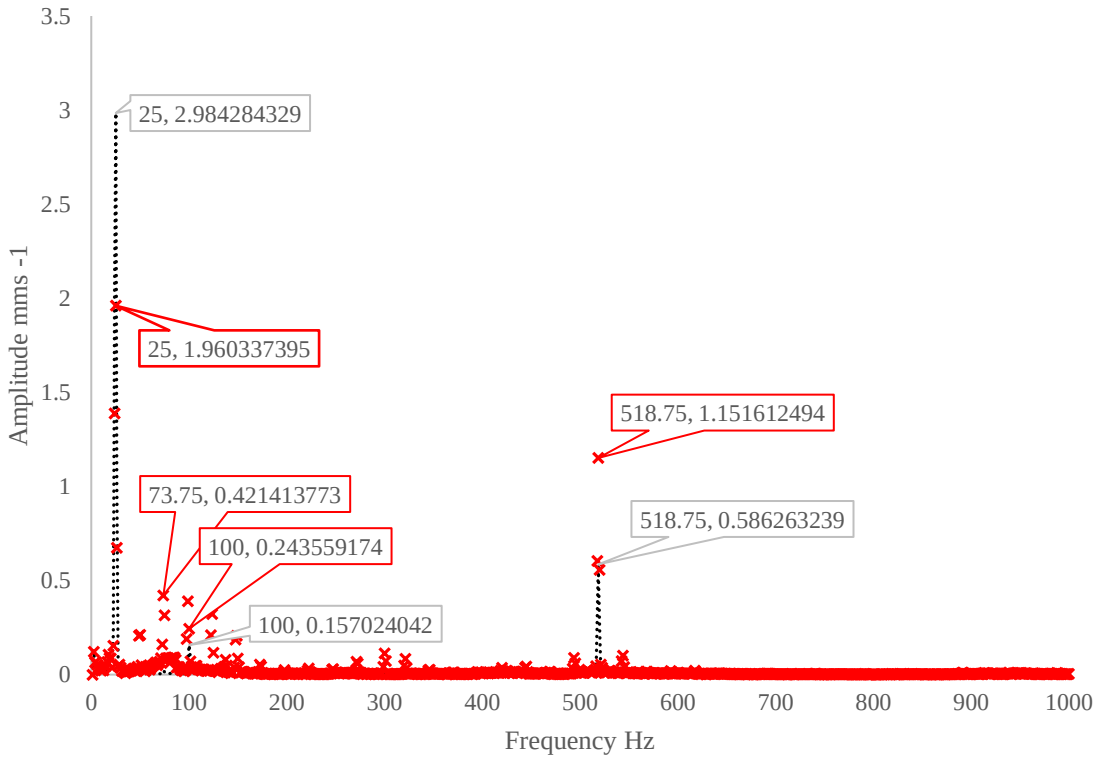
8	Figure E-7: Loose Mounting of Pump Foot - Velocity FFT



9	Figure E-8: Static Imbalance in Shaft - Velocity FFT



10	Figure E-9: Offset Misalignment in Shaft - Velocity FFT

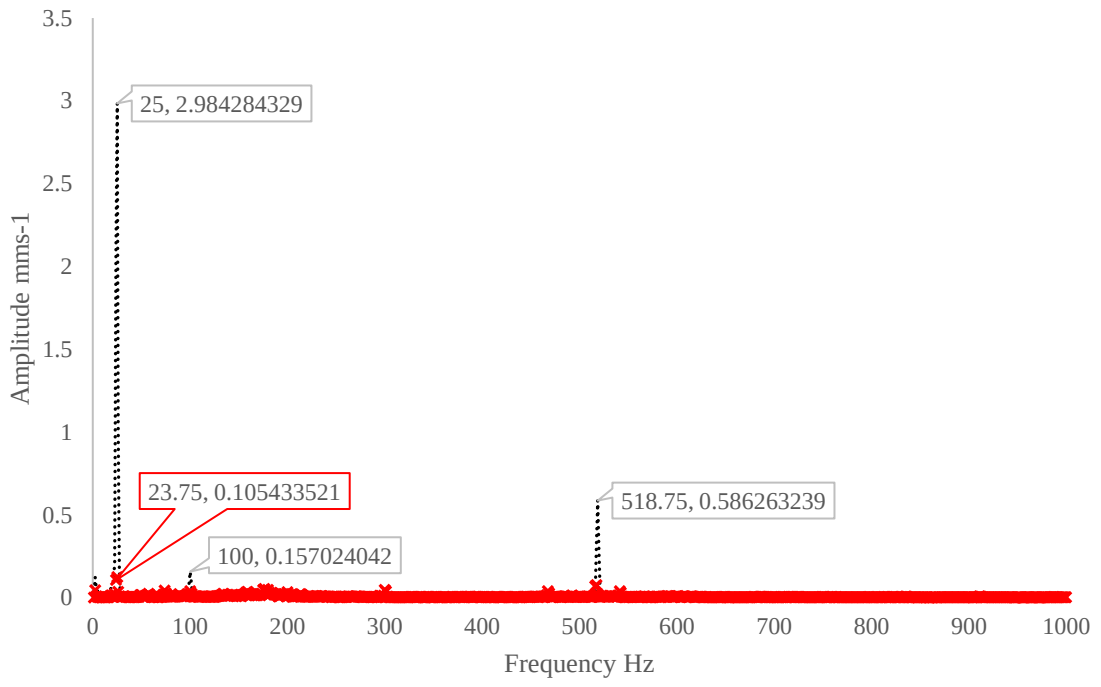


E.1.2 CM Data

The Number 2 General Service Pump CM configuration as described in Chapter 3, Section 3.5.2 was used to obtain all data displayed within this section.

In the following Figure E-10, the calculated values for f_0 and BPF are 25 Hz and 520 Hz respectively.

CM #2	**Figure E-10: Condition Monitoring of Number 2 General Service Pump - Velocity FFT Completed: 27 March 2018**



E.1.3 Velocity FFT Summary

A summary of the features extracted from the data presented in Figures E-1 to E-10 are shown in Table E-1. The table highlights that a combination of features is required to distinguish each of the classes. These features were described previously in Section 3.7.1.

Table E-1: Features Extracted from Velocity FFT Data Examples

	Feature												
	BPF	f_0	$3f_0$	$4f_0$	$5f_0$	$6f_0$	$7f_0$	$8f_0$	$12f_0$	$20f_0$	$36f_0$	$37f_0$	$37f_0$
Vessel Alongside, No Engines Running	0.5863	2.9843	0.0312	0.0226	0.1570	0.0128	0.0134	0.0031	0.0038	0.0122	0.0351	0.0051	0.0043
Vessel Alongside, Engines Running	0.5225	3.1792	0.0311	0.0226	0.0355	0.0434	0.0141	0.0188	0.0067	0.0268	0.0341	0.0053	0.0051
Vessel at Sea	0.2788	3.5117	0.1172	0.0214	0.0421	0.0217	0.0241	0.0116	0.0041	0.0230	0.0438	0.0065	0.0068
Worn Impeller	0.7125	4.9277	0.0178	0.0265	0.2844	0.0430	0.0292	0.0092	0.0173	0.0273	0.0514	0.0124	0.0153
Loose Packing	1.4118	3.6069	0.2282	0.1395	0.1243	0.0143	0.0916	0.0147	0.0083	0.0356	0.0610	0.0059	0.0069
Damaged Pump Drive-End Bearing	1.0604	5.9063	0.0368	0.0387	0.0479	0.0270	0.0332	0.0094	0.0051	0.0389	0.0281	0.0087	0.0192
Worn Pump Drive-End Bearing	0.2608	0.2789	0.3386	0.7151	0.0678	0.0716	0.0429	0.0470	0.0627	0.0364	0.0349	0.0082	0.0064
Static Imbalance in Shaft	1.7495	6.8416	0.0604	0.0355	0.0985	0.0487	0.0855	0.0123	0.0051	0.0870	0.0511	0.0064	0.0083
Offset Misalignment in Shaft	1.1516	1.9603	0.2125	0.4214	0.3910	0.3222	0.2070	0.0538	0.0240	0.1133	0.0899	0.0117	0.0103
Loose Mounting	0.9810	4.2312	0.3356	0.1524	0.1610	0.0234	0.0866	0.0065	0.0081	0.0326	0.1013	0.0110	0.0082
CM #2	0.0761	0.1238	0.0149	0.0420	0.0381	0.0047	0.0132	0.0461	0.0336	0.0457	0.0135	0.0080	0.0025

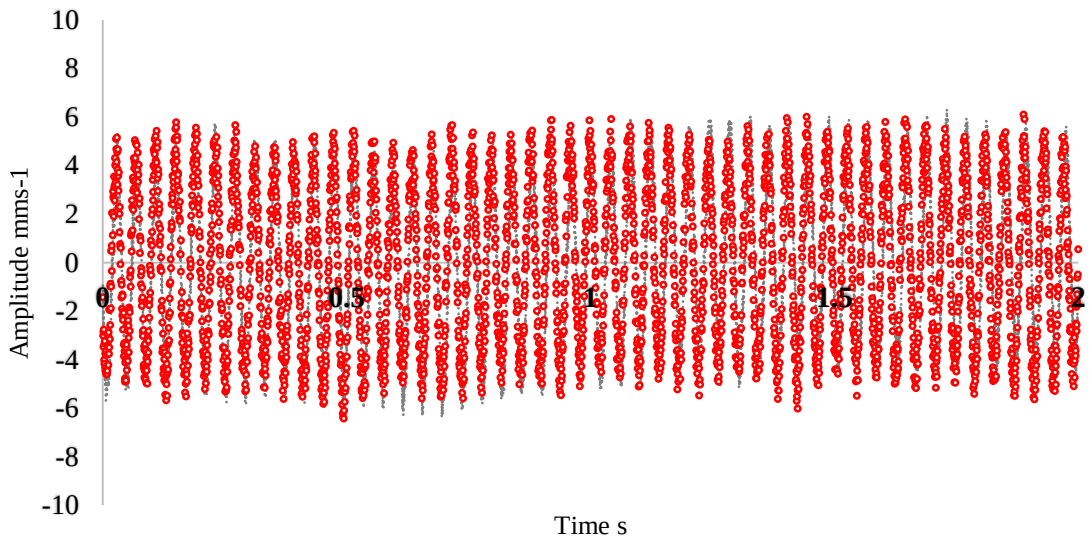
E.2 Velocity Waveform Data

E.2.1 Experimental Data

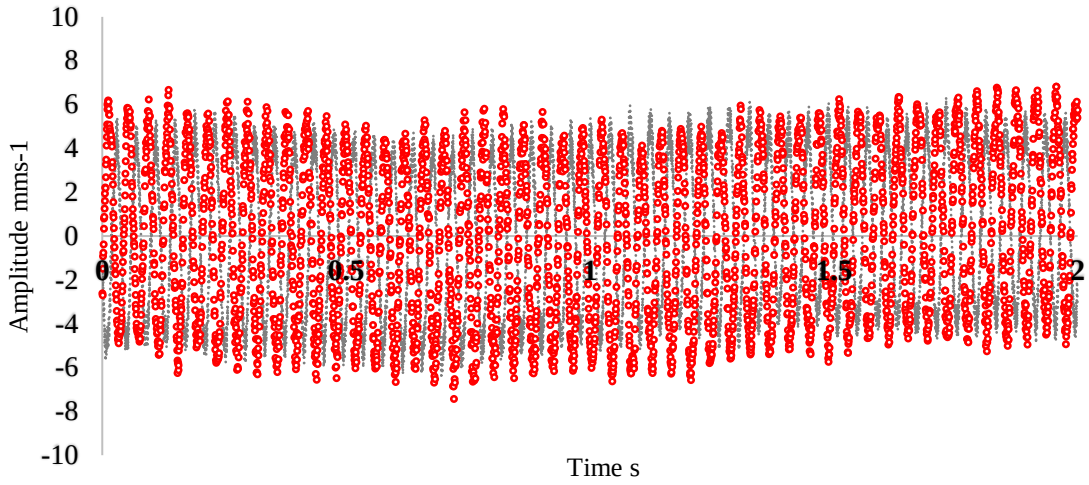
The Test pump configuration as described in Chapter 3, was used to obtain all experimental data displayed within this section in Figures E-11 to E-19.

The Dataset "1" class, 'Vessel Alongside, No Engines Running, OK to Operate' is plotted in grey for comparison in the following waveform Figures.

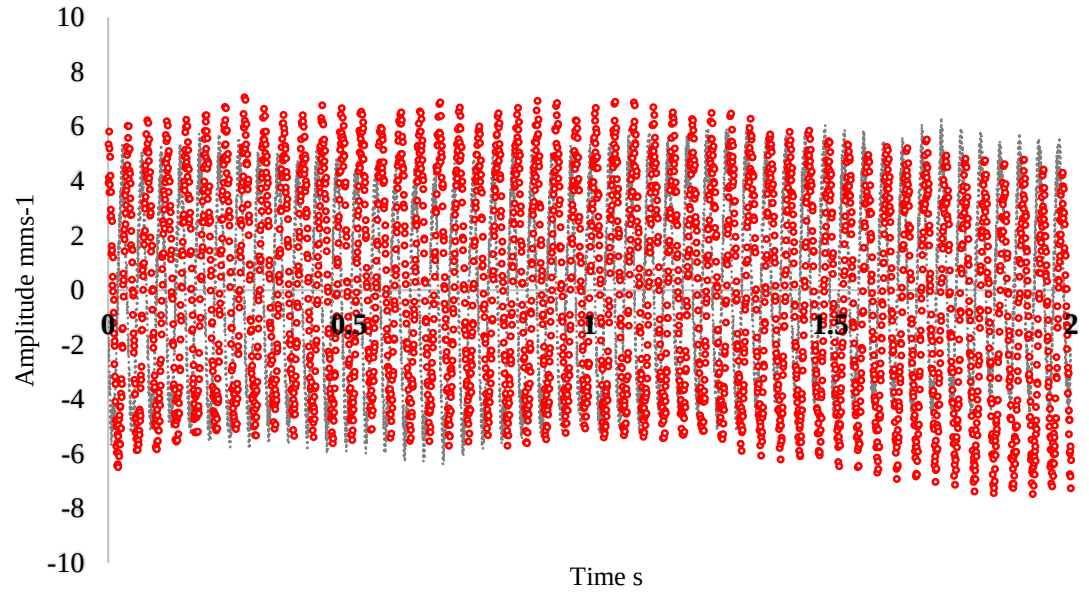
2	Figure E-11: Vessel Alongside, Engines Running - Velocity Waveform



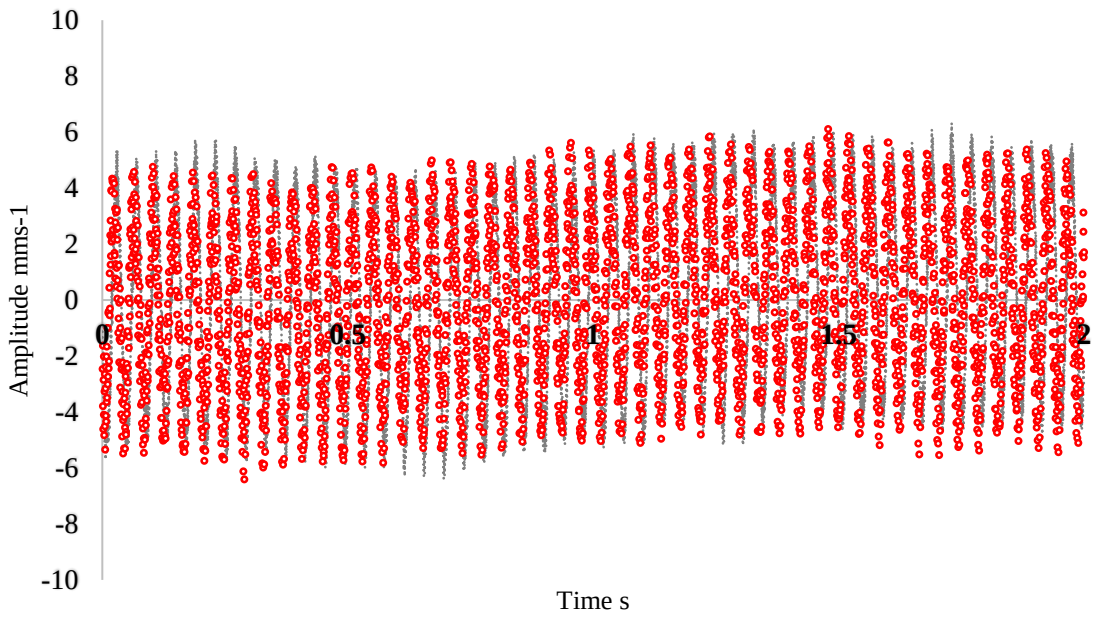
3	Figure E-12: Vessel at Sea - Velocity Waveform



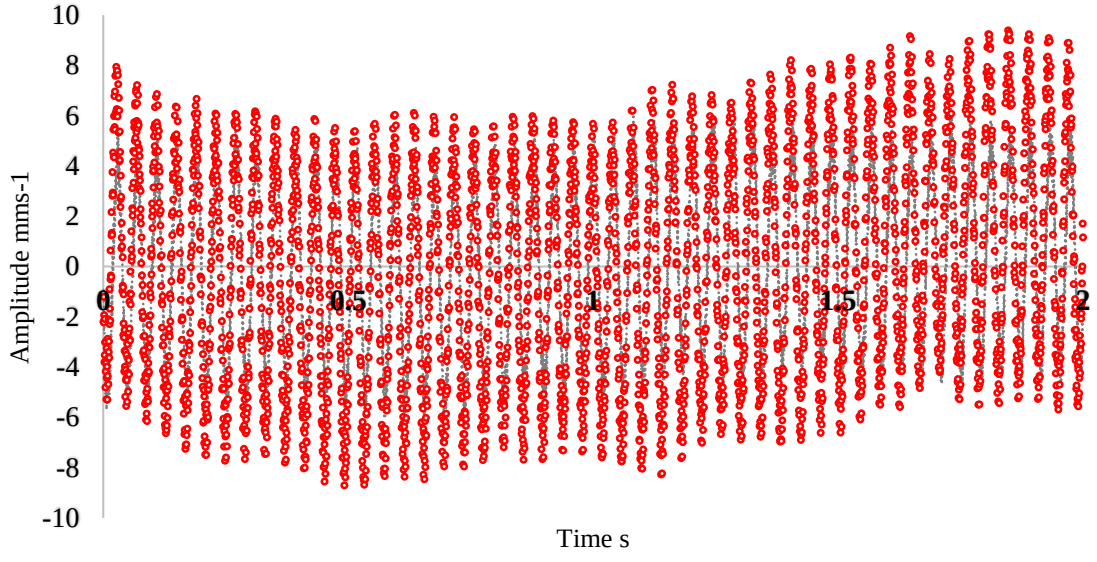
4	Figure E-13: Worn Impeller - Velocity Waveform



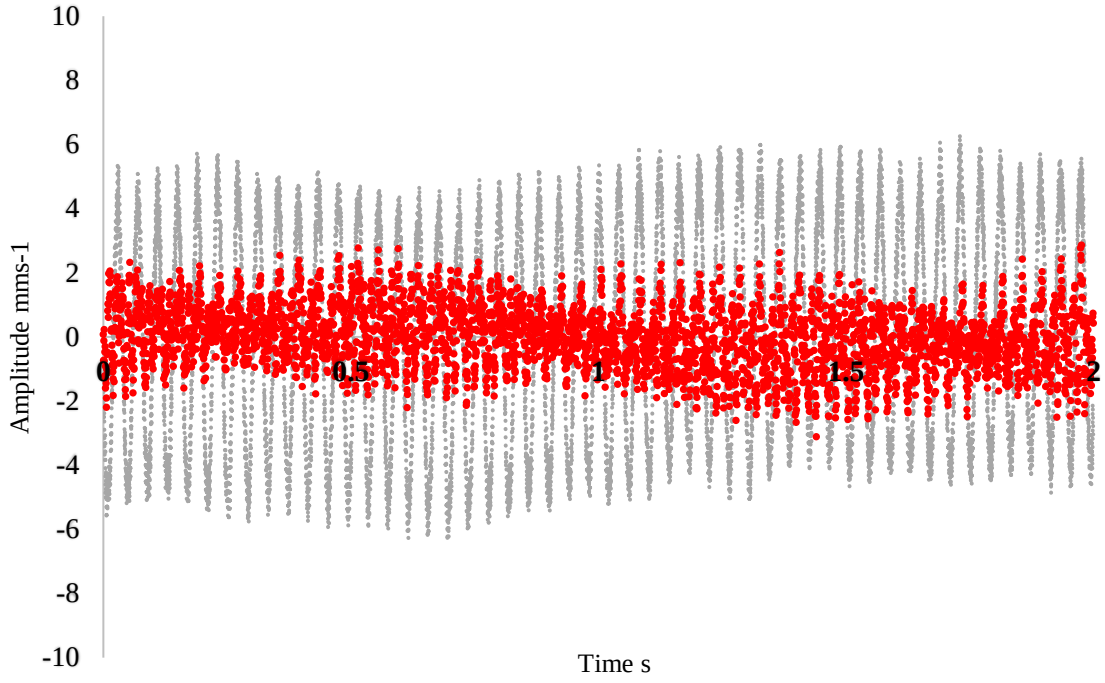
5	Figure E-14: Loose Packing - Velocity Waveform



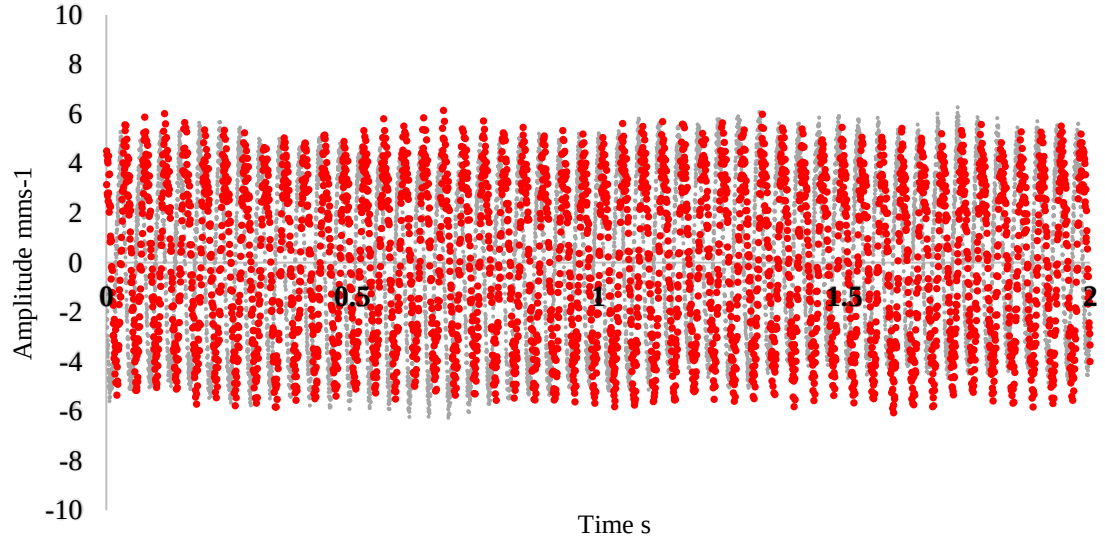
6	Figure E-15: Damaged Pump Drive-End Bearing - Velocity Waveform



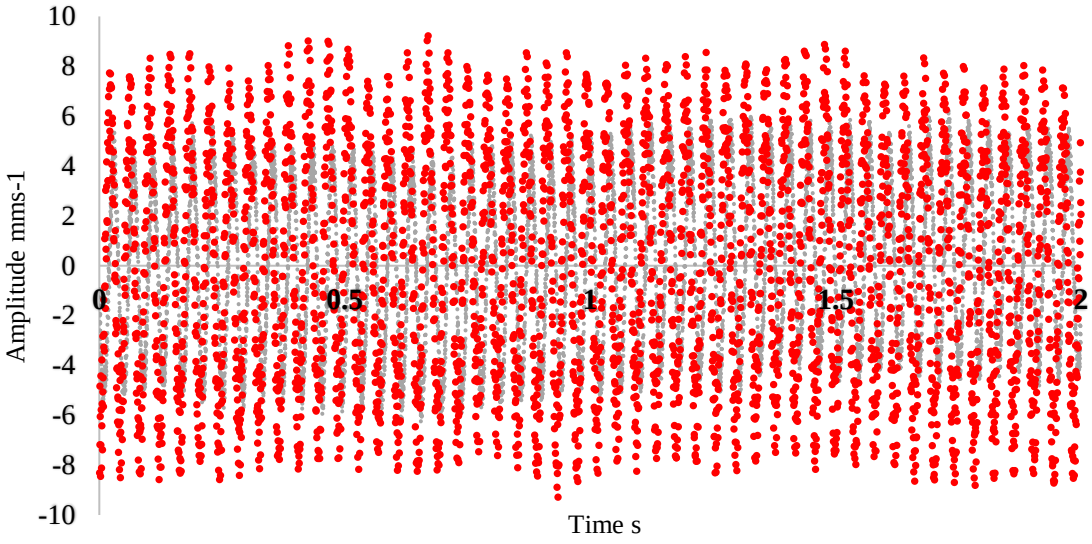
7	Figure E-16: Worn Pump Drive-End Bearing - Velocity Waveform



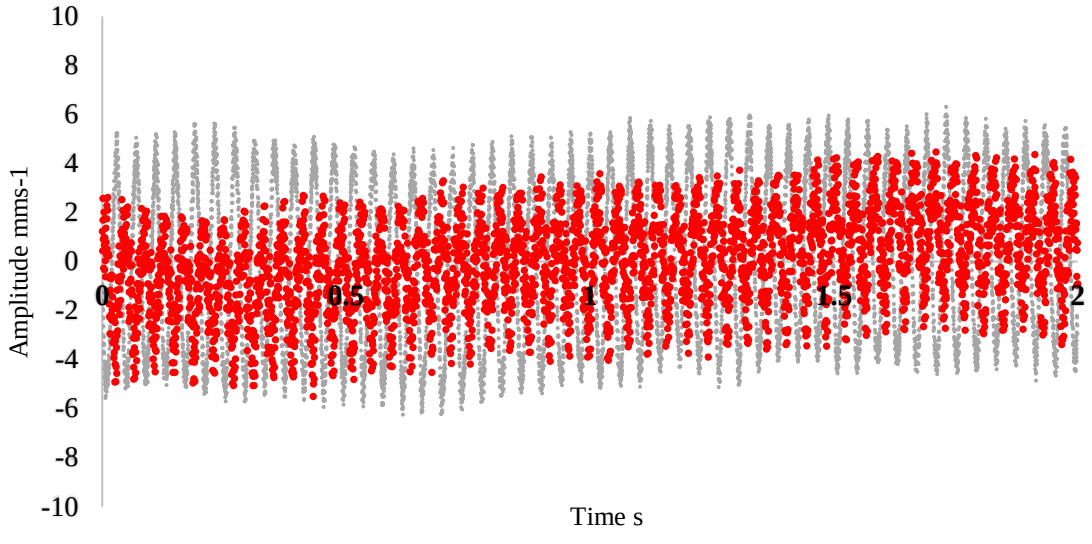
8	Figure E-17: Loose Mounting of Pump Foot - Velocity Waveform



9	Figure E-18: Static Imbalance in Shaft - Velocity Waveform



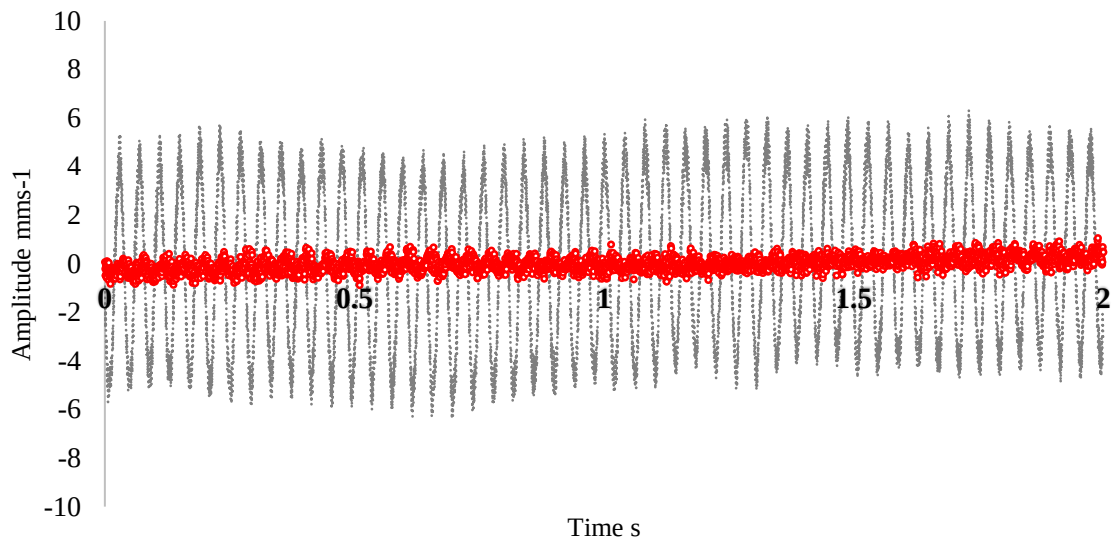
10	Figure E-19: Offset Misalignment in Shaft - Velocity Waveform



E.2.2 CM Data

The Number 2 General Service Pump CM configuration as described in Chapter 3 was used to obtain all data displayed within this section.

CM #2	**Figure E-20: Condition Monitoring of Number 2 General Service Pump - Velocity Waveform, Completed: 27 March 2018**



E.2.3 Velocity Waveform Summary

The features extracted from the velocity waveform data presented in Figures E-11 to E-20 are shown below in Table E-2. The table shows that the different classes may be distinguished using the mean, kurtosis and Sum features. These features were described previously in Section 3.7.1.

Table E-2: Features Extracted from Example Velocity Waveform Data

	Feature										
	Mean	**Std. Dev**	**Std. Error**	**Median**	**Variance**	**Skewness**	**Kurtosis**	**Range**	**Min**	**Max**	**Sum**
Vessel Alongside, Engines Running	0.0001	0.1304	0.0020	0.0003	0.0170	0.0683	2.0439	0.6349	-0.3300	0.3050	0.3591
Vessel Alongside, No Engines Running	-0.0002	0.1354	0.0021	0.0094	0.0183	0.0962	2.0195	0.5879	-0.2745	0.3134	-0.6390
Vessel at Sea	0.0004	0.0981	0.0015	0.0019	0.0096	0.0053	2.7670	0.6311	-0.3406	0.2905	1.7761
Worn Impeller	-0.0006	0.1367	0.0021	-0.0026	0.0187	0.0179	2.0204	0.6044	-0.3046	0.2998	-2.4570
Loose Packing	0.0002	0.2234	0.0035	-0.0006	0.0499	-0.0215	1.6568	0.8456	-0.4267	0.4189	0.9780
Damaged Pump Drive-End Bearing	0.0002	0.1742	0.0027	-0.0019	0.0303	-0.0391	1.8933	0.7368	-0.3743	0.3626	0.6412
Worn Pump Drive-End Bearing	0.0000	0.0812	0.0013	-0.0008	0.0066	0.0179	2.4280	0.4709	-0.2284	0.2425	-0.0450
Static Imbalance in Shaft	0.0005	0.2850	0.0045	-0.0028	0.0812	0.0975	1.7440	1.1219	-0.5267	0.5952	2.2343
Offset Misalignment in Shaft	-0.0002	0.1786	0.0028	-0.0056	0.0319	-0.0047	1.7766	0.7824	-0.4159	0.3665	-0.7072
Loose Mounting	-0.0003	0.1594	0.0025	-0.0052	0.0254	0.0558	1.8648	0.6797	-0.3224	0.3573	-1.1441
CM #2	0.0000	0.0309	0.0005	0.0006	0.0010	-0.0408	2.9079	0.2219	-0.1167	0.1052	0.0521

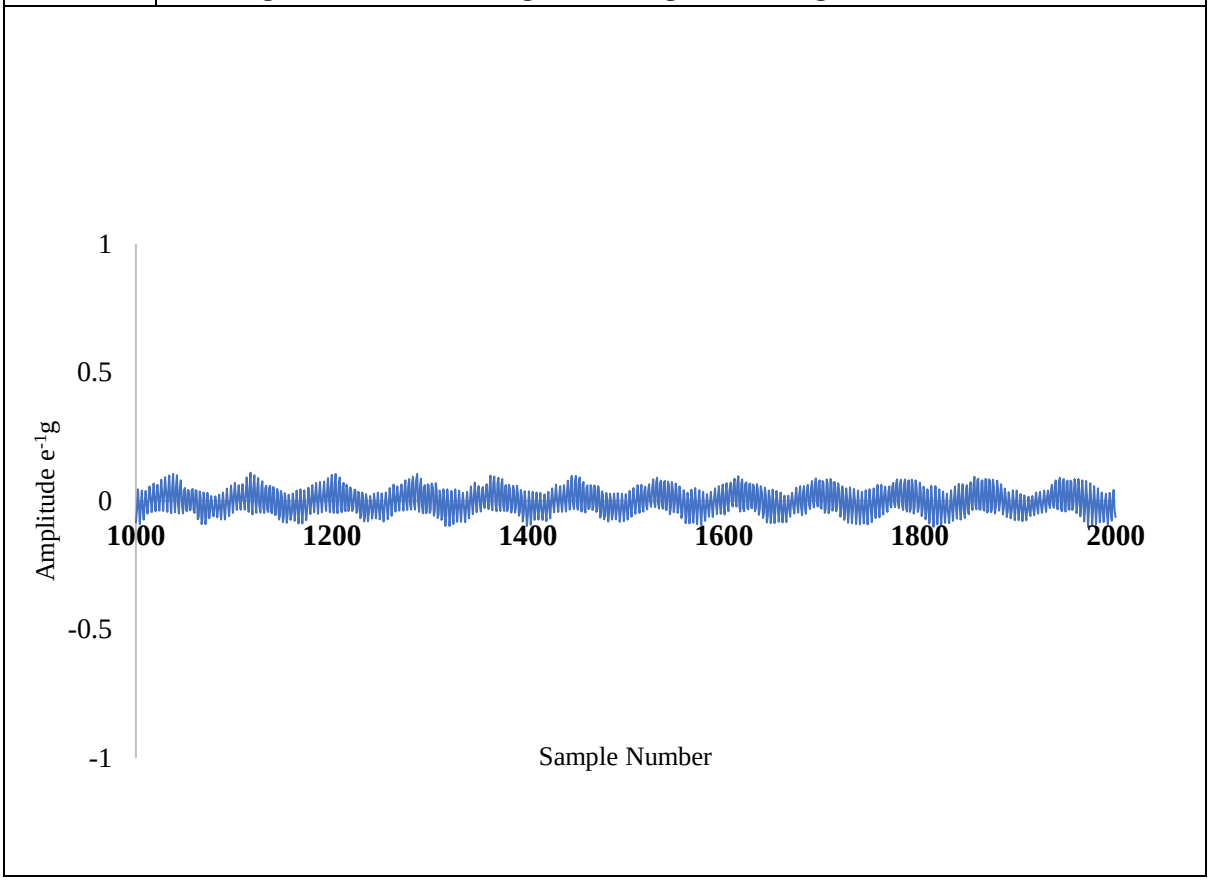
E.3 Acceleration Waveform Data

Acceleration waveform data are presented in this section for comparison with future similar studies. 1000 samples are presented in each Figure analogously to Sakthivel, Sugumaran, and Babudevasenapati (2010). Although some studies have been identified which conduct similar investigations (Kamiel, 2015; Mohanty, Pradhan, Mahalik, & Dastidar, 2012; Sakthivel et al., 2010) their results could not be compared to the present results as they were obtained under different experimental conditions. Design of the present experimental conditions was discussed in Chapter 3.

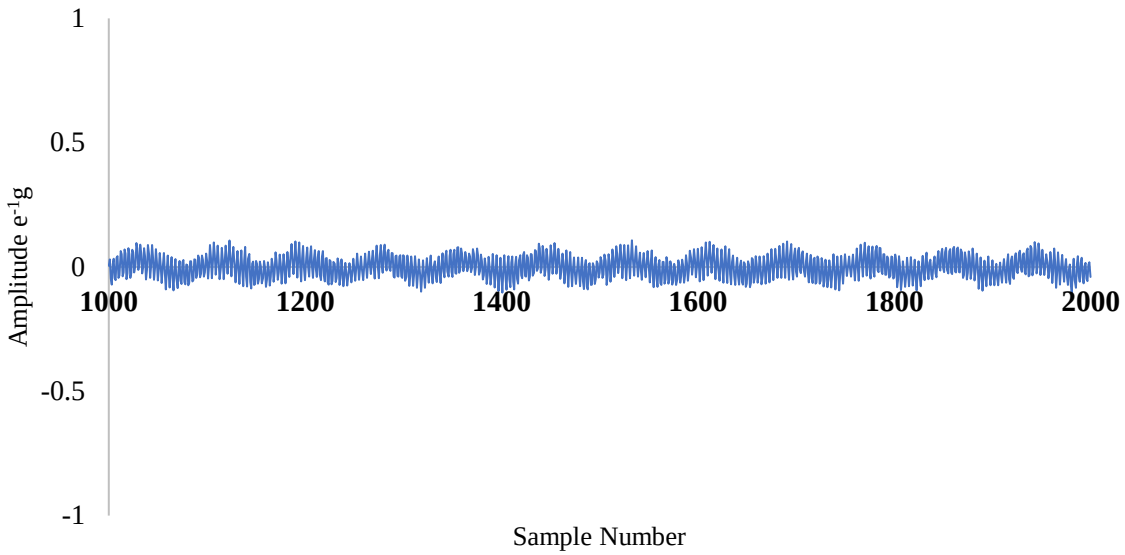
E.3.1 Experimental Data

The Test pump configuration as described in Chapter 3 was used to obtain all experimental data displayed within this section in Figures E-21 to E-30.

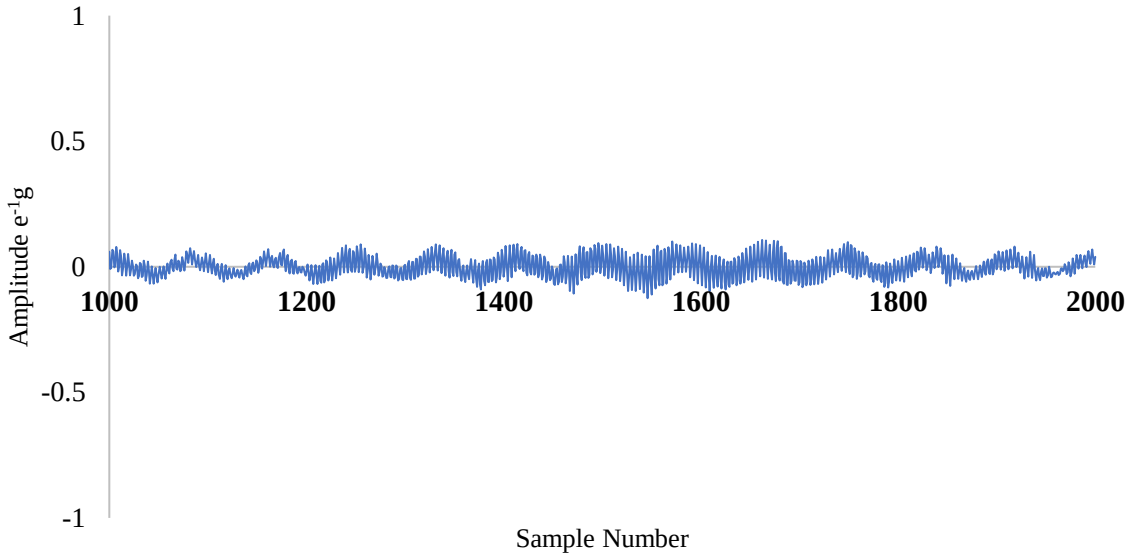
2	**Figure E-21: Vessel Alongside, No Engines Running - Acceleration Waveform**



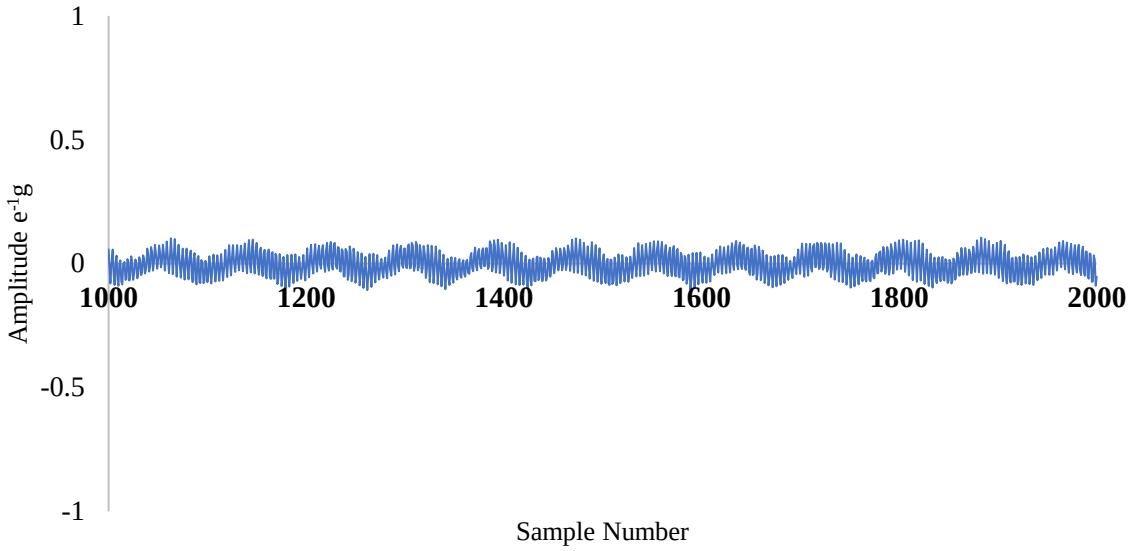
3	Figure E-22: Vessel Alongside, Engines Running - Acceleration Waveform



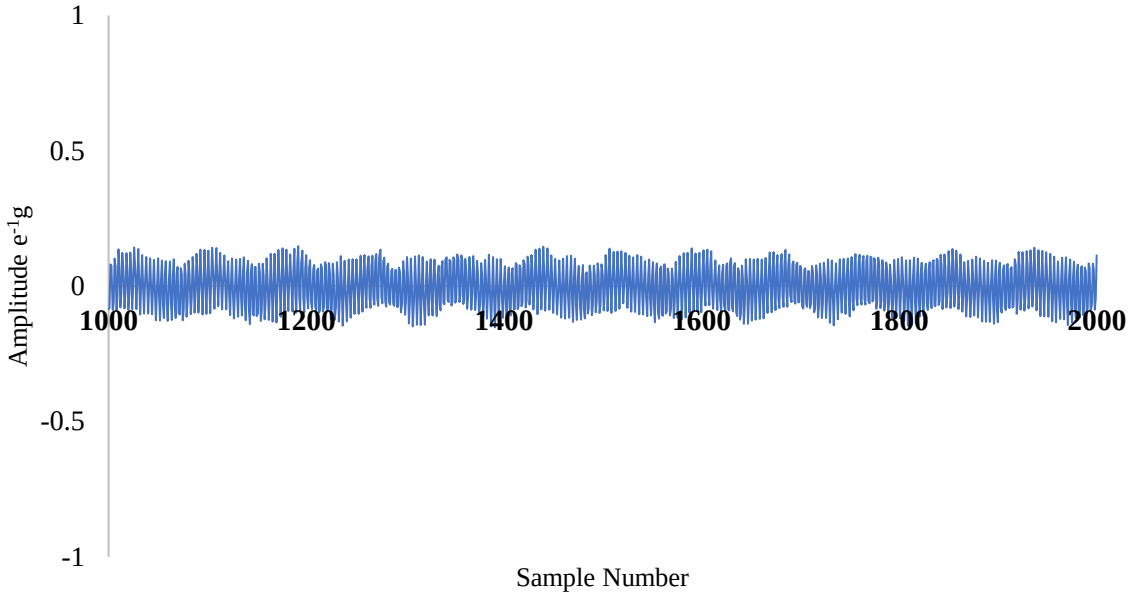
4	Figure E-23: Vessel at Sea - Acceleration Waveform



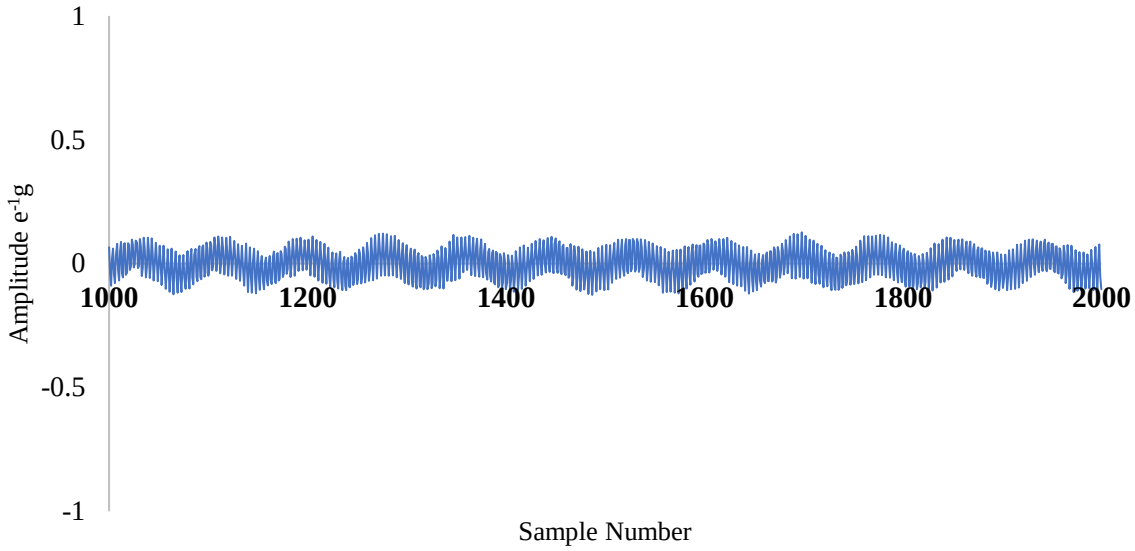
4	Figure E-24: Worn Impeller - Acceleration Waveform



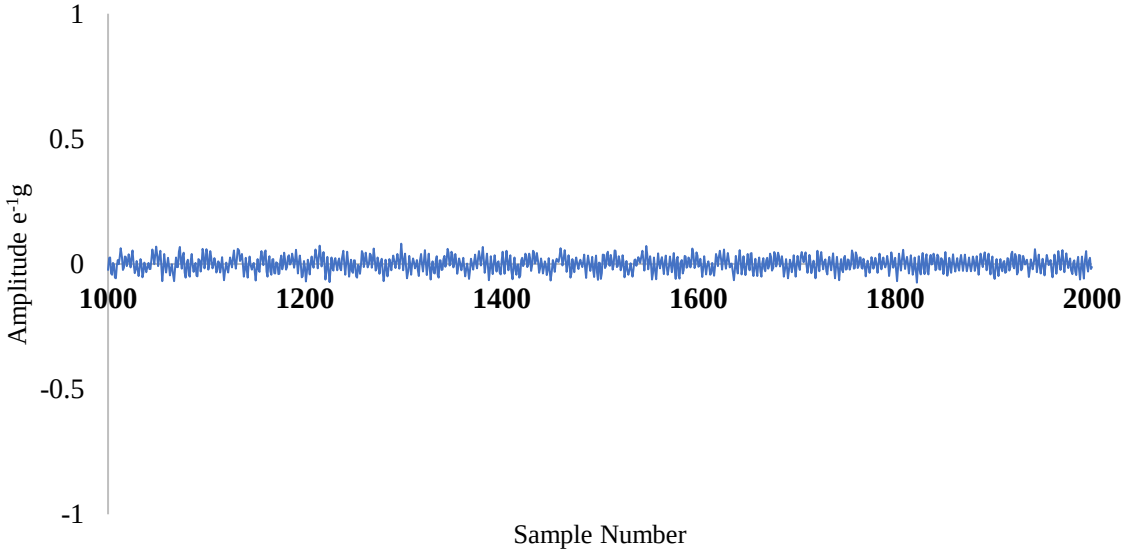
5	Figure E-25: Loose Packing - Acceleration Waveform



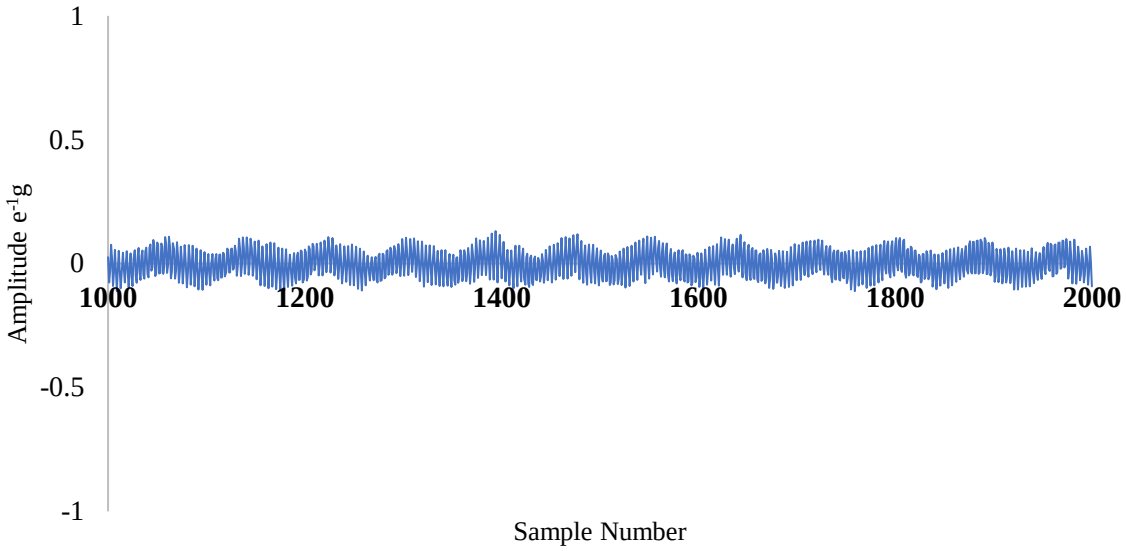
6	Figure E-26: Damaged Pump Drive-End Bearing - Acceleration Waveform



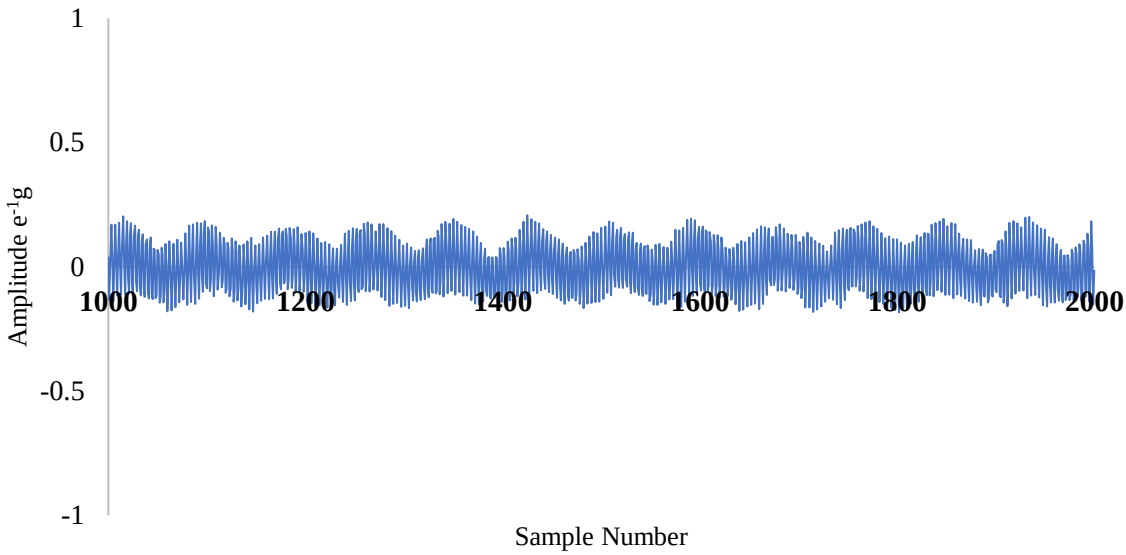
7	Figure E-27: Worn Pump Drive-End Bearing - Acceleration Waveform



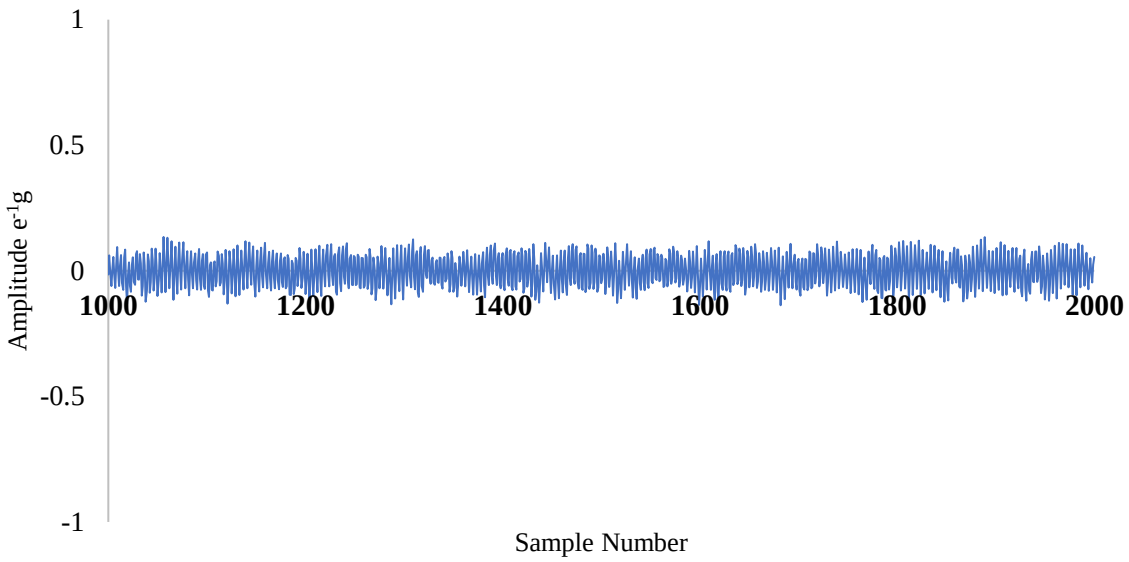
8	Figure E-28: Loose Mounting of Pump Foot - Acceleration Waveform



9	Figure E-29: Static Imbalance in Shaft - Acceleration Waveform



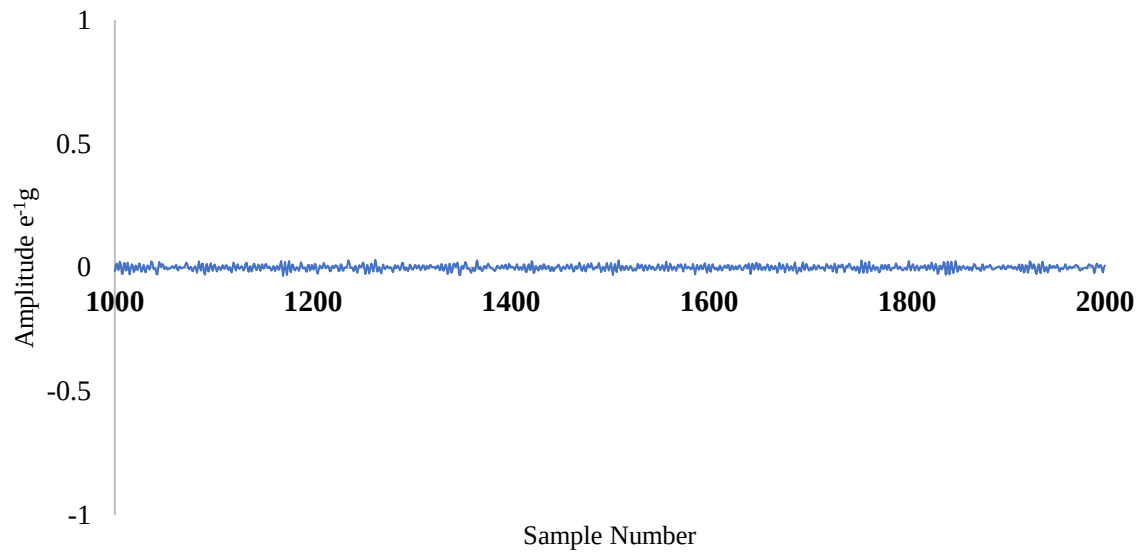
10	Figure E-30: Offset Misalignment in Shaft - Acceleration Waveform



E.3.2 CM Data

The Number 2 General Service Pump CM configuration as described in Chapter 3 was used to obtain all data displayed within this section.

CM #2	**Figure E-31: Condition Monitoring of Number 2 General Service Pump - Acceleration Waveform, Completed: 27 March 2018**



Appendix E References

Kamiel, B. P. (2015). Vibration-based multi-fault diagnosis for centrifugal pumps. *Department of Mechanical Engineering*.

Mohanty, A. R., Pradhan, P. K., Mahalik, N. P., & Dastidar, S. G. (2012). Fault detection in a centrifugal pump using vibration and motor current signature analysis. *International Journal of Automation and Control*, *6*(3-4), 261-276.

Sakthivel, N., Sugumaran, V., & Babudevasenapati, S. (2010). Vibration based fault diagnosis of monoblock centrifugal pump using decision tree. *Expert Systems with Applications*, *37*(6), 4040-4049.

APPENDIX F - Classification of data with random Gaussian Noise

The present Appendix E provides the result matrices describing the classification of 140 samples of data using classifiers C1, C2 and C3 which was used to develop Table 4-7 in Chapter 4. The data were summarised within the table.

F.1 Classification Results of C1

Linear discriminant classifier C1, developed previously in Chapter 4, describes the situation where the vessel is alongside and the engine room is quiet. C1 was used to classify 140 samples of learning data which were augmented with three levels of random gaussian noise, which had variances of 10%, 50% and 100%. The resulting confusion, certainty and doubt matrices of these classifications are shown in Figures F-1 to F-9.

10 % Random Gaussian Noise Variance

100	0	0	0	0	0	0	0
0	100	0	0	0	0	0	0
0	0	100	0	0	0	0	0
0	0	0	100	0	0	0	0
0	0	0	0	100	0	0	0
0	0	0	0	0	100	0	0
0	0	0	0	0	0	100	0
0	0	0	0	0	0	0	100

Figure F-1: C1 10% GN Confusion

100	NaN	NaN	NaN	NaN	NaN	NaN	NaN
NaN	100	NaN	NaN	NaN	NaN	NaN	NaN
NaN	NaN	100	NaN	NaN	NaN	NaN	NaN
NaN	NaN	NaN	100	NaN	NaN	NaN	NaN
NaN	NaN	NaN	NaN	100	NaN	NaN	NaN
NaN	NaN	NaN	NaN	NaN	100	NaN	NaN
NaN	NaN	NaN	NaN	NaN	NaN	100	NaN
NaN	NaN	NaN	NaN	NaN	NaN	NaN	100

Figure F-2: C1 10% GN Certainty

2.46E-61	NaN	NaN	NaN	NaN	NaN	NaN	NaN
NaN	2.41E-127	NaN	NaN	NaN	NaN	NaN	NaN
NaN	NaN	8.45E-21	NaN	NaN	NaN	NaN	NaN
NaN	NaN	NaN	1.34E-55	NaN	NaN	NaN	NaN
NaN	NaN	NaN	NaN	0	NaN	NaN	NaN
NaN	NaN	NaN	NaN	NaN	8.22E-20	NaN	NaN
NaN	NaN	NaN	NaN	NaN	NaN	9.49E-278	NaN
NaN	NaN	NaN	NaN	NaN	NaN	NaN	0

Figure F-3: C1 10% GN Doubt

50 % Random Gaussian Noise Variance

58	18	0	12	2	10	0	0
20	60	0	20	0	0	0	0
0	0	90	0	0	10	0	0
0	0	0	90	0	10	0	0
20	0	0	0	80	0	0	0
10	0	0	10	0	70	10	0
0	10	0	10	0	0	80	0
0	0	0	0	0	0	0	100

Figure F-4: C1 50% GN Confusion

99.99	97.32	NaN	100	100	99.86	NaN	NaN
100	100	NaN	100	NaN	NaN	NaN	NaN
NaN	NaN	99.99	NaN	NaN	100	NaN	NaN
NaN	NaN	NaN	100	NaN	100	NaN	NaN
100	NaN	NaN	NaN	100	NaN	NaN	NaN
100	NaN	NaN	100	NaN	100	100	NaN
NaN	100	NaN	100	NaN	NaN	100	NaN
NaN	NaN	NaN	NaN	NaN	NaN	NaN	100

Figure F-5: C1 50% GN Certainty

5.30E-04	2.68E+00	NaN	1.48E-42	8.38E-35	5.16E-05	NaN	NaN
1.74E-135	5.28E-31	NaN	4.35E-159	NaN	NaN	NaN	NaN
NaN	NaN	5.74E-03	NaN	NaN	6.39E-28	NaN	NaN
NaN	NaN	NaN	1.00E-43	NaN	2.69E-36	NaN	NaN
1.28E-56	NaN	NaN	NaN	5.77E-27	NaN	NaN	NaN
3.92E-159	NaN	NaN	6.38E-262	NaN	2.51E-11	0	NaN
NaN	1.22E-211	NaN	0	NaN	NaN	1.85E-32	NaN
NaN	NaN	NaN	NaN	NaN	NaN	NaN	1.54E-200

Figure F-6: C1 50% GN Doubt

100 % Random Gaussian Noise Variance

18	14	10	6	20	16	16	0
10	40	10	20	0	0	20	0
0	0	60	0	0	30	10	0
0	20	0	50	10	10	10	0
10	20	10	0	60	0	0	0
0	0	30	0	0	70	0	0
0	0	0	10	10	0	80	0
0	0	0	0	0	0	10	90

Figure F-7: C1 100% GN Confusion

100	100	96.56	100	100	100	100	NaN
100	100	100	98.86	NaN	NaN	100	NaN
NaN	NaN	100	NaN	NaN	100	100	NaN
NaN	100	NaN	100	100	100	100	NaN
100	100	100	NaN	100	NaN	NaN	NaN
NaN	NaN	100	NaN	NaN	100	NaN	NaN
NaN	NaN	NaN	100	100	NaN	100	NaN
NaN	NaN	NaN	NaN	NaN	NaN	100	100

Figure F-8: C1 100% GN Certainty

1.82E-08	5.54E-24	1.56E-20	1.20E-68	1.44E-25	1.46E-20	2.66E-303	NaN
1.11E-30	4.68E-197	0	1.14	NaN	NaN	0	NaN
NaN	NaN	3.57E-09	NaN	NaN	1.55E-21	0	NaN
NaN	8.96E-130	NaN	8.08E-53	0	1.01E-42	1.31E-107	NaN
4.46E-229	0	0	NaN	5.36E-244	NaN	NaN	NaN
NaN	NaN	8.29E-10	NaN	NaN	4.90E-19	NaN	NaN
NaN	NaN	NaN	0	0	NaN	1.02E-67	NaN
NaN	NaN	NaN	NaN	NaN	NaN	5.04E-132	1.69E-35

Figure F-9: C1 100% GN Doubt

F.2 Classification Results of C2

Linear discriminant classifier C2 developed previously in Chapter 4, describes the situation where the vessel is alongside and the main engines are operational. C2 was used to classify 140 samples of learning data which were augmented with three levels of random gaussian noise, which had variances of 10%, 50% and 100%. The resulting confusion, certainty and doubt matrices of these classifications are shown in Figures F-10 to F-18.

10 % Random Gaussian Noise Variance

100	0	0	0	0	0	0	0
0	100	0	0	0	0	0	0
0	0	100	0	0	0	0	0
0	0	0	100	0	0	0	0
0	0	0	0	100	0	0	0
0	0	0	0	0	100	0	0
0	0	0	0	0	0	100	0
0	0	0	0	0	0	0	100

Figure F-10: C2 10% GN Confusion

100	NaN	NaN	NaN	NaN	NaN	NaN	NaN
NaN	100	NaN	NaN	NaN	NaN	NaN	NaN
NaN	NaN	100	NaN	NaN	NaN	NaN	NaN
NaN	NaN	NaN	100	NaN	NaN	NaN	NaN
NaN	NaN	NaN	NaN	100	NaN	NaN	NaN
NaN	NaN	NaN	NaN	NaN	100	NaN	NaN
NaN	NaN	NaN	NaN	NaN	NaN	100	NaN
NaN	NaN	NaN	NaN	NaN	NaN	NaN	100

Figure F-11: C2 10% GN Certainty

2.17E-73	NaN	NaN	NaN	NaN	NaN	NaN	NaN
NaN	4.14E-74	NaN	NaN	NaN	NaN	NaN	NaN
NaN	NaN	2.61E-11	NaN	NaN	NaN	NaN	NaN
NaN	NaN	NaN	1.03E-62	NaN	NaN	NaN	NaN
NaN	NaN	NaN	NaN	8.66E-305	NaN	NaN	NaN
NaN	NaN	NaN	NaN	NaN	1.79E-10	NaN	NaN
NaN	NaN	NaN	NaN	NaN	NaN	1.13E-140	NaN
NaN	NaN	NaN	NaN	NaN	NaN	NaN	3.04E-255

Figure F-12: C2 10% GN Doubt

50 % Random Gaussian Noise Variance

40	10	0	30	0	20	0	0
10	60	0	20	0	0	10	0
0	0	80	0	0	20	0	0
0	10	0	70	0	20	0	0
0	0	10	0	90	0	0	0
30	0	0	20	0	50	0	0
0	0	0	0	0	0	100	0
0	0	0	0	0	0	0	100

Figure F-13: C2 50% GN Confusion

99.99	100	NaN	100	NaN	99.99	NaN	NaN
100	100	NaN	99.99	NaN	NaN	100	NaN
NaN	NaN	96.43	NaN	NaN	98.77	NaN	NaN
NaN	100	NaN	100	NaN	100	NaN	NaN
NaN	NaN	99.99	NaN	100	NaN	NaN	NaN
100	NaN	NaN	100	NaN	96.35	NaN	NaN
NaN	NaN	NaN	NaN	NaN	NaN	100	NaN
NaN	NaN	NaN	NaN	NaN	NaN	NaN	100

Figure F-14: C2 50% GN Certainty

4.67E-07	2.14E-77	NaN	9.35E-12	NaN	1.35E-71	NaN	NaN
2.08E-209	1.80E-10	NaN	2.43E-59	NaN	NaN	1.06E-15	NaN
NaN	NaN	3.57	NaN	NaN	1.23E+00	NaN	NaN
NaN	2.56E-89	NaN	8.80E-34	NaN	9.28E-86	NaN	NaN
NaN	NaN	6.02E-72	NaN	3.72E-265	NaN	NaN	NaN
4.15E-19	NaN	NaN	5.95E-36	NaN	3.65E+00	NaN	NaN
NaN	NaN	NaN	NaN	NaN	NaN	3.31E-92	NaN
NaN	NaN	NaN	NaN	NaN	NaN	NaN	4.35E-166

Figure F-15: C2 50% GN Doubt

100 % Random Gaussian Noise Variance

10	10	10	30	30	0	10	0
10	20	0	10	0	20	40	0
0	0	60	10	10	0	10	10
10	10	0	20	0	40	20	0
0	0	20	0	70	10	0	0
30	10	0	10	0	40	10	0
0	10	0	20	0	0	70	0
0	10	0	0	0	0	0	90

Figure F-16: C2 100% GN Confusion

100	100	100	99.99	88.53	NaN	100	NaN
100	99.99	NaN	100	NaN	100	100	NaN
NaN	NaN	100	100	100	NaN	100	100
100	99.99	NaN	100	NaN	99.99	100	NaN
NaN	NaN	99.99	NaN	99.98	97.75	NaN	NaN
100	100	NaN	100	NaN	100	100	NaN
NaN	99.99	NaN	100	NaN	NaN	100	NaN
NaN	100	NaN	NaN	NaN	NaN	NaN	100

Figure F-17: C2 100% GN Certainty

6.25E-131	9.62E-225	0.00E+00	6.20E-102	1.15E+01	NaN	0.00E+00	NaN
3.85E-300	2.41E-07	NaN	7.24E-276	NaN	0	6.70E-65	NaN
NaN	NaN	3.95E-13	5.12E-199	0	NaN	0	7.23E-194
8.65E-285	6.29E-07	NaN	2.67E-14	NaN	2.05E-49	9.68E-79	NaN
NaN	NaN	4.87E-205	NaN	2.12E-02	1.40E-150	NaN	NaN
5.31E-68	2.96E-252	NaN	0	NaN	4.54E-15	0	NaN
NaN	4.12E-05	NaN	1.85E-228	NaN	NaN	4.20E-54	NaN
NaN	3.67E-117	NaN	NaN	NaN	NaN	NaN	8.92E-42

Figure F-18: C2 100% GN Doubt

F.3 Classification Results of C3 and C4

Linear discriminant classifier C3, developed previously in Chapter 4, describes the situation where the vessel is in motion at sea, slow steaming around the harbour at <4 knots in the absence of an emergency. Classifier C4 describes the situation where the vessel is in motion at sea, slow steaming around the harbour at <4 knots in an emergency. There is no difference between the learning samples used to develop C3 and C4, thus their classification results are equivalent. However, two classifiers exist to aid in future model if the learning data in C4 must be changed.

C3 was used to classify 140 samples of learning data which were augmented with three levels of random gaussian noise, which had variances of 10%, 50% and 100%. The resulting confusion, certainty and doubt matrices of these classifications are shown in Figure F-19 to F-27.

10 % Random Gaussian Noise Variance

100	0	0	0	0	0	0	0
0	100	0	0	0	0	0	0
0	0	100	0	0	0	0	0
0	0	0	100	0	0	0	0
0	0	0	0	100	0	0	0
0	0	0	0	0	100	0	0
0	0	0	0	0	0	100	0
0	0	0	0	0	0	0	100

Figure F-19: C3/C4 10% GN Confusion

100	NaN	NaN	NaN	NaN	NaN	NaN	NaN
NaN	100	NaN	NaN	NaN	NaN	NaN	NaN
NaN	NaN	100	NaN	NaN	NaN	NaN	NaN
NaN	NaN	NaN	100	NaN	NaN	NaN	NaN
NaN	NaN	NaN	NaN	100	NaN	NaN	NaN
NaN	NaN	NaN	NaN	NaN	100	NaN	NaN
NaN	NaN	NaN	NaN	NaN	NaN	100	NaN
NaN	NaN	NaN	NaN	NaN	NaN	NaN	100

Figure F-20: C3/C4 10% GN Certainty

3.64E-119	NaN	NaN	NaN	NaN	NaN	NaN	NaN
NaN	1.34E-126	NaN	NaN	NaN	NaN	NaN	NaN
NaN	NaN	2.38E-16	NaN	NaN	NaN	NaN	NaN
NaN	NaN	NaN	5.24E-123	NaN	NaN	NaN	NaN
NaN	NaN	NaN	NaN	0	NaN	NaN	NaN
NaN	NaN	NaN	NaN	NaN	5.08E-11	NaN	NaN
NaN	NaN	NaN	NaN	NaN	NaN	1.67E-166	NaN
NaN	NaN	NaN	NaN	NaN	NaN	NaN	2.44E-303

Figure F-21: C3/C4 10% GN Doubt

50 % Random Gaussian Noise Variance

30	30	0	20	10	10	0	0
30	30	0	10	0	0	30	0
0	10	70	10	10	0	0	0
0	20	0	80	0	0	0	0
0	0	0	0	100	0	0	0
40	0	10	20	0	30	0	0
0	10	0	10	0	0	80	0
0	10	0	0	0	0	0	90

Figure F-22: C3/C4 50% GN Confusion

100	100	NaN	100	100	100	NaN	NaN
100	100	NaN	100	NaN	NaN	100	NaN
NaN	100	100	100	100	NaN	NaN	NaN
NaN	100	NaN	99.99	NaN	NaN	NaN	NaN
NaN	NaN	NaN	NaN	100	NaN	NaN	NaN
100	NaN	100	100	NaN	99.99	NaN	NaN
NaN	100	NaN	99.99	NaN	NaN	100	NaN
NaN	100	NaN	NaN	NaN	NaN	NaN	100

Figure F-23: C3/C4 50% GN Certainty

6.54E-26	1.04E-21	NaN	9.36E-117	1.00E-50	0.00E+00	NaN	NaN
6.20E-110	2.03E-46	NaN	0	NaN	NaN	5.44E-23	NaN
NaN	0	6.40E-11	0	1.34E-33	NaN	NaN	NaN
NaN	8.21E-179	NaN	6.64E-06	NaN	NaN	NaN	NaN
NaN	NaN	NaN	NaN	5.05E-266	NaN	NaN	NaN
8.81E-11	NaN	1.47E-41	1.47E-90	NaN	1.23E-02	NaN	NaN
NaN	0.00E+00	NaN	2.64E-123	NaN	NaN	1.17E-35	NaN
NaN	2.30E-81	NaN	NaN	NaN	NaN	NaN	7.27E-57

Figure F-24: C3/C4 50% GN Doubt

100 % Random Gaussian Noise Variance

0	20	20	40	20	0	0	0
20	10	10	30	20	10	0	0
0	20	40	0	10	20	10	0
0	20	0	40	10	10	20	0
10	0	10	0	80	0	0	0
20	0	40	40	0	0	0	0
0	10	20	0	0	0	70	0
10	0	0	30	0	10	0	50

Figure F-25: C3/C4 100% GN Confusion

NaN	100	100	99.99	100	NaN	NaN	NaN
100	100	100	100	100	100	NaN	NaN
NaN	100	100	NaN	100	100	100	NaN
NaN	100	NaN	100	99.97	100	100	NaN
100	NaN	100	NaN	100	NaN	NaN	NaN
100	NaN	100	100	NaN	NaN	NaN	NaN
NaN	100	100	NaN	NaN	NaN	100	NaN
100	NaN	NaN	100	NaN	100	NaN	100

Figure F-26: C3/C4 100% GN Certainty

NaN	1.24E-233	3.51E-282	2.13E-50	4.21E-303	NaN	NaN	NaN
9.73E-235	1.00E-226	0	1.00E-15	0	0	NaN	NaN
NaN	0	1.63E-43	NaN	0	2.51E-39	0	NaN
NaN	4.20E-85	NaN	1.03E-90	0	0	4.66E-254	NaN
0	NaN	0	NaN	3.79E-59	NaN	NaN	NaN
6.42E-89	NaN	5.48E-25	8.49E-101	NaN	NaN	NaN	NaN
NaN	2.30E-36	0	NaN	NaN	NaN	1.46E-35	NaN
1.48E-36	NaN	NaN	1.25E-22	NaN	3.29E-106	NaN	5.48E-41

Figure F-27: C3/C4 100% GN Doubt

APPENDIX G - Qualitative Description of Consequences used in Decision Analysis

The qualitative descriptions of the decision consequences interpreted into a MAU function in Chapter 4 are described in Table G-1. These were developed in accordance with the decision trees with expert assistance. Two contexts are considered; the vessel alongside and the vessel moving at sea. The additional contexts of 'engines running' and 'emergency' did not affect the qualitative descriptions of the consequences. The states θ_1 to θ_8 were described previously in Chapter 4, Table 4-10.

Table G-1: Qualitative Description of Consequences

Context	Action	Description	Branch	Consequence
Alongside	A_1	Stop No. 2 General Service Pump to check inside	Repair time > 30 mins / θ_1	Changeover to No. 1 General Service Pump. Perform maintenance IMMEDIATELY on No. 2 General Service Pump
Alongside	A_1	Stop No. 2 General Service Pump to check inside	Repair time < 30 mins / θ_1	Perform maintenance IMMEDIATELY on No. 2 General Service Pump, Restart No. 2 General Service Pump
Alongside	A_1	Stop No. 2 General Service Pump to check inside	Repair time > 30 mins / θ_2	Changeover to No. 1 General Service Pump. Perform maintenance IMMEDIATELY on No. 2 General Service Pump
Alongside	A_1	Stop No. 2 General Service Pump to check inside	Repair time < 30 mins / θ_2	Perform maintenance IMMEDIATELY on No. 2 General Service Pump, Restart No. 2 General Service Pump
Alongside	A_1	Stop No. 2 General Service Pump to check inside	Repair time > 30 mins / θ_3	Changeover to No. 1 General Service Pump. Perform maintenance IMMEDIATELY on No. 2 General Service Pump
Alongside	A_1	Stop No. 2 General Service Pump to check inside	Repair time < 30 mins / θ_3	Perform maintenance IMMEDIATELY on No. 2 General Service Pump, Restart No. 2 General Service Pump
Alongside	A_1	Stop No. 2 General Service Pump to check inside	Repair time > 30 mins / θ_4	Changeover to No. 1 General Service Pump. Perform maintenance IMMEDIATELY on No. 2 General Service Pump
Alongside	A_1	Stop No. 2 General Service Pump to check inside	Repair time < 30 mins / θ_4	Perform maintenance IMMEDIATELY on No. 2 General Service Pump, Restart No. 2 General Service Pump
Alongside	A_1	Stop No. 2 General Service Pump to check inside	Repair time > 30 mins / θ_5	Changeover to No. 1 General Service Pump. Perform maintenance IMMEDIATELY on No. 2 General Service Pump
Alongside	A_1	Stop No. 2 General Service Pump to check inside	Repair time < 30 mins / θ_5	Perform maintenance IMMEDIATELY on No. 2 General Service Pump, Restart No. 2 General Service Pump
Alongside	A_1	Stop No. 2 General Service Pump to check inside	Repair time > 30 mins / θ_6	Changeover to No. 1 General Service Pump. Perform maintenance IMMEDIATELY on No. 2 General Service Pump
Alongside	A_1	Stop No. 2 General Service Pump to check inside	Repair time < 30 mins / θ_6	Perform maintenance IMMEDIATELY on No. 2 General Service Pump, Restart No. 2 General Service Pump
Alongside	A_1	Stop No. 2 General Service Pump to check inside	Repair time > 30 mins / θ_7	Changeover to No. 1 General Service Pump. Perform maintenance IMMEDIATELY on No. 2 General Service Pump
Alongside	A_1	Stop No. 2 General Service Pump to check inside	Repair time < 30 mins / θ_7	Perform maintenance IMMEDIATELY on No. 2 General Service Pump, Restart No. 2 General Service Pump

Context	Action	Description	Branch	Consequence
Alongside	A_1	Stop No. 2 General Service Pump to check inside	Repair time > 30 mins / θ_8	Changeover to No. 1 General Service Pump. Perform maintenance IMMEDIATELY on No. 2 General Service Pump
Alongside	A_1	Stop No. 2 General Service Pump to check inside	Repair time < 30 mins / θ_8	Perform maintenance IMMEDIATELY on No. 2 General Service Pump, Restart No. 2 General Service Pump
At Sea	A_1	Stop No. 2 General Service Pump to check inside, changeover to No.1 General Service Pump immediately	θ_1	Allow to No.1 General Service Pump to finish job, INSPECT No.2 General Service Pump ASAP
At Sea	A_1	Stop No. 2 General Service Pump to check inside, changeover to No.1 General Service Pump immediately	θ_2	Allow to No.1 General Service Pump to finish job, SCHEDULE maintenance on No.2 General Service Pump ASAP
At Sea	A_1	Stop No. 2 General Service Pump to check inside, changeover to No.1 General Service Pump immediately	θ_3	Allow to No.1 General Service Pump to finish job, SCHEDULE maintenance on No.2 General Service Pump ASAP
At Sea	A_1	Stop No. 2 General Service Pump to check inside, changeover to No.1 General Service Pump immediately	θ_4	Allow to No.1 General Service Pump to finish job, SCHEDULE maintenance on No.2 General Service Pump ASAP
At Sea	A_1	Stop No. 2 General Service Pump to check inside, changeover to No.1 General Service Pump immediately	θ_5	Allow to No.1 General Service Pump to finish job, SCHEDULE maintenance on No.2 General Service Pump ASAP
At Sea	A_1	Stop No. 2 General Service Pump to check inside, changeover to No.1 General Service Pump immediately	θ_6	Allow to No.1 General Service Pump to finish job, SCHEDULE maintenance on No.2 General Service Pump ASAP

Context	Action	Description	Branch	Consequence
At Sea	A_1	Stop No. 2 General Service Pump to check inside, changeover to No.1 General Service Pump immediately	θ_7	Allow to No.1 General Service Pump to finish job, SCHEDULE maintenance on No.2 General Service Pump ASAP
At Sea	A_1	Stop No. 2 General Service Pump to check inside, changeover to No.1 General Service Pump immediately	θ_8	Allow to No.1 General Service Pump to finish job, SCHEDULE maintenance on No.2 General Service Pump ASAP
Both	A_2	Inspect No. 2 General Service Pump more than once in 6 hours	Stops / θ_1	Stop No. 2 General Service Pump, Changeover to No. 1 General Service Pump, SCHEDULE maintenance on No. 2 General Service Pump
Both	A_2	Inspect No. 2 General Service Pump more than once in 6 hours	Stops / θ_2	Stop No. 2 General Service Pump, Changeover to No. 1 General Service Pump, SCHEDULE maintenance on No. 2 General Service Pump
Both	A_2	Inspect No. 2 General Service Pump more than once in 6 hours	Stops / θ_3	Stop No. 2 General Service Pump, Changeover to No. 1 General Service Pump, SCHEDULE maintenance on No. 2 General Service Pump
Both	A_2	Inspect No. 2 General Service Pump more than once in 6 hours	Stops / θ_4	Stop No. 2 General Service Pump, Changeover to No. 1 General Service Pump, SCHEDULE maintenance on No. 2 General Service Pump
Both	A_2	Inspect No. 2 General Service Pump more than once in 6 hours	Stops / θ_5	Stop No. 2 General Service Pump, Changeover to No. 1 General Service Pump, SCHEDULE maintenance on No. 2 General Service Pump
Both	A_2	Inspect No. 2 General Service Pump more than once in 6 hours	Stops / θ_6	Stop No. 2 General Service Pump, Changeover to No. 1 General Service Pump, SCHEDULE maintenance on No. 2 General Service Pump
Both	A_2	Inspect No. 2 General Service Pump more than once in 6 hours	Stops / θ_7	Stop No. 2 General Service Pump, Changeover to No. 1 General Service Pump, SCHEDULE maintenance on No. 2 General Service Pump

Context	Action	Description	Branch	Consequence
Both	A_2	Inspect No. 2 General Service Pump more than once in 6 hours	Stops / θ_8	Stop No. 2 General Service Pump, Changeover to No. 1 General Service Pump, SCHEDULE maintenance on No. 2 General Service Pump
Both	A_3	Inspect No. 2 General Service Pump once in 6 hours	Stops / θ_1	Stop No. 2 General Service Pump, Changeover to No. 1 General Service Pump, SCHEDULE maintenance on No. 2 General Service Pump
Both	A_3	Inspect No. 2 General Service Pump once in 6 hours	Stops / θ_2	Stop No. 2 General Service Pump, Changeover to No. 1 General Service Pump, SCHEDULE maintenance on No. 2 General Service Pump
Both	A_3	Inspect No. 2 General Service Pump once in 6 hours	Stops / θ_3	Stop No. 2 General Service Pump, Changeover to No. 1 General Service Pump, SCHEDULE maintenance on No. 2 General Service Pump
Both	A_3	Inspect No. 2 General Service Pump once in 6 hours	Stops / θ_4	Stop No. 2 General Service Pump, Changeover to No. 1 General Service Pump, SCHEDULE maintenance on No. 2 General Service Pump
Both	A_3	Inspect No. 2 General Service Pump once in 6 hours	Stops / θ_5	Stop No. 2 General Service Pump, Changeover to No. 1 General Service Pump, SCHEDULE maintenance on No. 2 General Service Pump
Both	A_3	Inspect No. 2 General Service Pump once in 6 hours	Stops / θ_6	Stop No. 2 General Service Pump, Changeover to No. 1 General Service Pump, SCHEDULE maintenance on No. 2 General Service Pump
Both	A_3	Inspect No. 2 General Service Pump once in 6 hours	Stops / θ_7	Stop No. 2 General Service Pump, Changeover to No. 1 General Service Pump, SCHEDULE maintenance on No. 2 General Service Pump
Both	A_3	Inspect No. 2 General Service Pump once in 6 hours	Stops / θ_8	Stop No. 2 General Service Pump, Changeover to No. 1 General Service Pump, SCHEDULE maintenance on No. 2 General Service Pump

Context	Action	Description	Branch	Consequence
Both	A_4	Inspect No. 2 General Service Pump once in 12 hours	Stops / θ_1	Stop No. 2 General Service Pump, Changeover to No. 1 General Service Pump, SCHEDULE maintenance on No. 2 General Service Pump
Both	A_4	Inspect No. 2 General Service Pump once in 12 hours	Stops / θ_2	Stop No. 2 General Service Pump, Changeover to No. 1 General Service Pump, SCHEDULE maintenance on No. 2 General Service Pump
Both	A_4	Inspect No. 2 General Service Pump once in 12 hours	Stops / θ_3	Stop No. 2 General Service Pump, Changeover to No. 1 General Service Pump, SCHEDULE maintenance on No. 2 General Service Pump
Both	A_4	Inspect No. 2 General Service Pump once in 12 hours	Stops / θ_4	Stop No. 2 General Service Pump, Changeover to No. 1 General Service Pump, SCHEDULE maintenance on No. 2 General Service Pump
Both	A_4	Inspect No. 2 General Service Pump once in 12 hours	Stops / θ_5	Stop No. 2 General Service Pump, Changeover to No. 1 General Service Pump, SCHEDULE maintenance on No. 2 General Service Pump
Both	A_4	Inspect No. 2 General Service Pump once in 12 hours	Stops / θ_6	Stop No. 2 General Service Pump, Changeover to No. 1 General Service Pump, SCHEDULE maintenance on No. 2 General Service Pump
Both	A_4	Inspect No. 2 General Service Pump once in 12 hours	Stops / θ_7	Stop No. 2 General Service Pump, Changeover to No. 1 General Service Pump, SCHEDULE maintenance on No. 2 General Service Pump
Both	A_4	Inspect No. 2 General Service Pump once in 12 hours	Stops / θ_8	Stop No. 2 General Service Pump, Changeover to No. 1 General Service Pump, SCHEDULE maintenance on No. 2 General Service Pump
Both	A_5	Inspect No. 2 General Service Pump once in 24 hours (Do nothing)	Stops / θ_1	Stop No. 2 General Service Pump, Changeover to No. 1 General Service Pump, SCHEDULE maintenance on No. 2 General Service Pump

Context	Action	Description	Branch	Consequence
Both	A_5	Inspect No. 2 General Service Pump once in 24 hours (Do nothing)	Stops / θ_2	Stop No. 2 General Service Pump, Changeover to No. 1 General Service Pump, SCHEDULE maintenance on No. 2 General Service Pump
Both	A_5	Inspect No. 2 General Service Pump once in 24 hours (Do nothing)	Stops / θ_3	Stop No. 2 General Service Pump, Changeover to No. 1 General Service Pump, SCHEDULE maintenance on No. 2 General Service Pump
Both	A_5	Inspect No. 2 General Service Pump once in 24 hours (Do nothing)	Stops / θ_4	Stop No. 2 General Service Pump, Changeover to No. 1 General Service Pump, SCHEDULE maintenance on No. 2 General Service Pump
Both	A_5	Inspect No. 2 General Service Pump once in 24 hours (Do nothing)	Stops / θ_5	Stop No. 2 General Service Pump, Changeover to No. 1 General Service Pump, SCHEDULE maintenance on No. 2 General Service Pump
Both	A_5	Inspect No. 2 General Service Pump once in 24 hours (Do nothing)	Stops / θ_6	Stop No. 2 General Service Pump, Changeover to No. 1 General Service Pump, SCHEDULE maintenance on No. 2 General Service Pump
Both	A_5	Inspect No. 2 General Service Pump once in 24 hours (Do nothing)	Stops / θ_7	Stop No. 2 General Service Pump, Changeover to No. 1 General Service Pump, SCHEDULE maintenance on No. 2 General Service Pump
Both	A_5	Inspect No. 2 General Service Pump once in 24 hours (Do nothing)	Stops / θ_8	Stop No. 2 General Service Pump, Changeover to No. 1 General Service Pump, SCHEDULE maintenance on No. 2 General Service Pump

APPENDIX H - Measuring Attribute Values

The present Appendix H outlines the equations applied or values assigned to measure each value of attribute y_a where $a = 1...6$ in all $\vec{y}$ for further evaluation (Refer to Chapter 4 and software in Appendix C). Qualitative descriptions of each outcome $\vec{y}$ are provided in Appendix G.

Some y_a are calculated or assigned differently because of the decision context or action used to reach them. Consequence attributes and states of nature were defined in Chapter 4 as follows:

ATTRIBUTE y_a	**DESCRIPTION**
y_1	Downtime
y_2	Expected Repair cost of pump
y_3	Expected Repair cost of Vessel
y_4	Expected Number of people severely injured
y_5	Routine Maintenance Cost
y_6	Lloyds Compliance

CLASS NUMBER c_k	**STATE OF NATURE** θ_j	**DESCRIPTION**
1	θ_1	No fault / OK to operate
2	θ_2	Worn Impeller
3	θ_3	Loose Packing
4	θ_4	Damaged Pump Drive-End Bearing
5	θ_5	Worn Pump Drive-End Bearing
6	θ_6	Loose pump foot
7	θ_7	Static Imbalance in Shaft
8	θ_8	Offset Misalignment in Shaft

H.1 Context: Vessel Alongside, Only Pump Running

The expert decides between the five actions in Table H-1, which form the first decision tree.

Table H-1: Possible actions, vessel Alongside

NUMBER	ACTION
A_1	Stop No.2 Pump to check inside
A_2	Inspect pump more than once in 6 hours
A_3	Inspect pump after 6 hours
A_4	Inspect pump after 12 hours
A_5	Inspect pump after 24 hours / Do nothing

The outcomes of each of the actions are uncertain and so the actions are modelled as lotteries. The structure of the decision tree and these lotteries were described in Chapter 4. Determining expected utility solutions of the lotteries and tree requires evaluation of the multi-attribute utility of each consequence. This initially requires measurement of all attributes y_a where $a = 1...6$ in all consequences $\vec{y}$. The calculations or assignments used to obtain all y_a in the context where the vessel is alongside are tabulated and discussed in the present Section H.1.

Table H-2 shows the calculations or assignments used for y_a when the vessel is at the wharf. We adopt the lottery terminology from Chapter 3. The total number of inspections over the following 24 hours were included when calculating y_1 for all actions. This resulted in the multiple inspection times shown for $l(A_2)$ to $l(A_4)$ in Table H-2. y_2 to y_4 were assigned a zero value when θ_1 occurs across all lotteries in addition to the calculations in Table H-2 as θ_1 represents the normal operation condition. Also, the table shows that $y_6 = 1$ in all cases but this attribute must be included because $y_6 = 0$ in other decision contexts beyond the scope of the present work. A MATLAB script has been developed to perform the calculations or assignments shown in Table H-2, producing one 16 x 6 matrix and four 5 x 6 matrices. A subsequent function determines the multi-attribute utility of values in these matrices.

Table H-2: Vessel Alongside Attribute Values

ATTRIBUTE y_a	$L(A_1)$	$l(A_2)$	$l(A_3)$	$l(A_4)$	$l(A_5)$
y_1	$Mtn(\theta_j) + Extra + Penalty$	$(5 \times Insp) + Extra$	$(4 \times Insp) + Extra$	$(2 \times Insp) + Extra$	$Insp + Extra$
y_2	0	$P(\theta_j \mid \vec{x}) \times NewPump$	$\frac{P(\theta_j \mid \vec{x}) \times NewPump}{Correction^1}$	$\frac{P(\theta_j \mid \vec{x}) \times NewPump}{Correction^2}$	$\frac{P(\theta_j \mid \vec{x}) \times NewPump}{Correction^4}$
y_3	0	$P(\theta_j \mid \vec{x}) \times P(LossVessel \mid \theta_j) \times NewVessel$	$\frac{P(\theta_j \mid \vec{x}) \times P(LossVessel \mid \theta_j) \times NewVessel}{Correction^1}$	$\frac{P(\theta_j \mid \vec{x}) \times P(LossVessel \mid \theta_j) \times NewVessel}{Correction^2}$	$\frac{P(\theta_j \mid \vec{x}) \times P(LossVessel \mid \theta_j) \times NewVessel}{Correction^4}$
y_4	0	$P(\theta_j) \times P(Injury \mid \theta_j) \times NumCrew$	$P(\theta_j \mid \vec{x}) \times P(Injury \mid \theta_j) \times NumCrew$	$P(\theta_j \mid \vec{x}) \times P(Injury \mid \theta_j) \times NumCrew$	$P(\theta_j \mid \vec{x}) \times P(Injury \mid \theta_j) \times NumCrew$
y_5	$(y_1 \times EngRate) + Prt(\theta_j) + Lbr(\theta_j)$	$y_1 \times EngRate$	$\frac{y_1 \times EngRate}{Correction^1}$	$\frac{y_1 \times EngRate}{Correction^2}$	$\frac{y_1 \times EngRate}{Correction^4}$
y_6	1	1	1	1	1

The following variables in Table H-3 were used in Table H-2.

Table H-3: Variables used in Attribute Value Calculations

VARIABLE	DESCRIPTION	VALUE
$Mtn(\theta_j)$	The time in hours required to perform maintenance given state θ_j.	Refer to Table H-4
Extra	The additional time in hours required to stop the pump, perform an inspection, or switch pumps which does not include maintenance itself (captured in $Mtn(\theta_j)$	The sum of additional hours derived from the qualitative description of the consequence (Refer to Appendix G), where stopping the pump, performing an inspection or switching pumps each add 1/3 of an hour.
Penalty	The additional time in hours assumed to be incurred because the repair for θ_i will take longer than 0.5 hours (Refer to lotteries in Chapter 3).	0.1 hours
EngRate	Engineer's hourly salary in $AUD	$59
$Prt(\theta_j)$	The cost of parts needed to perform maintenance if state θ_j occurs in $AUD as quoted by the supplier in $AUD in June 2017.	Refer to Table H-4
$Lbr(\theta_j)$	The cost of additional labour (beyond one Engineer) to perform maintenance if state θ_j occurs in $AUD as estimated by the expert.	Refer to Table H-4
Insp	The time in hours needed to perform a non-invasive inspection of the pump.	1/3 hours
$P(\theta_j \mid \vec{x})$	The posterior probability of state θ_j given measurement $\vec{x}$ as determined by the relevant classifier.	No units

VARIABLE	DESCRIPTION	VALUE
$P(LossVessel \mid \theta_j)$	The subjective conditional likelihood that the vessel will sink if state θ_j occurs as estimated by the expert.	No units, refer to Table H-4
NewVessel	The cost of a new vessel equivalent to the vessel involved in the present study as estimated by the expert in $AUD in June 2017.	$10 million
$P\left(Injury \mid \theta_j\right)$	The subjective conditional likelihood of injury to nearby crew if state θ_j occurs as estimated by the expert.	Refer to Table H-4
NumCrew	The number of crew aboard the vessel.	Assuming a full crew of 4.
Correction	An assumed correction term to account for the time value of money (Net Present Value), applied to balance the values considering a 24-hour period.	1.006849315

The state-dependent in Table H-4 quantities were also incorporated into Table H-3. Subjective probability quantities $P(LossVessel \mid \theta_j)$ and $P\left(Injury \mid \theta_j\right)$ were estimated by the expert using a reference experiment as described in Chapter 4.

Table H-4: State Dependent Variable Values

	STATE OF NATURE θ_j							
VARIABLE	θ_1	θ_2	θ_3	θ_4	θ_5	θ_6	θ_7	θ_8
$Mtn\left(\theta_j\right)$	1/6	3	.5	2	2	4	4	0.5
$Prt\left(\theta_j\right)$	0	614.69	8.73	20.99	20.99	151.77	540.64	0
$Lbr\left(\theta_j\right)$	9.80	177.00	29.50	118.00	118.00	236.00	236.00	29.5.00
$P(LossVessel \mid \theta_j)$	0.01	0.02	0.03	0.03	0.03	0.1	0.2	1
$P\left(Injury \mid \theta_j\right)$	0.01	0.02	0.03	0.03	0.04	0.05	0.2	1

H.2: Context: Vessel Alongside, Engines Running

This decision tree possesses the same structure as the first context.

H.3 Context: Vessel at Sea, Not an Emergency

The expert decides between the five actions in Table H-5 which form the third decision tree.

Table H-5: Possible actions, vessel at sea

NUMBER	ACTION
A_1	Stop No. 2 Pump to check inside, change to No. 1 Pump immediately
A_2	Inspect pump more than once in 6 hours
A_3	Inspect pump after 6 hours
A_4	Inspect pump after 12 hours
A_5	Inspect pump after 24 hours / Do nothing

The actions are modelled as lotteries in Chapter 4. The present Section H.3. tabulates and discusses the calculations or assignments used to obtain all y_a where $a = 1...6$ in the context where the vessel is at sea.

Table H-6 shows the calculations or assignments used for y_a. The lottery terminology from Chapter 3, the variable definitions from Table H-3 and the state-specific variable values used in Table G-4 are retained. The total inspection time is calculated over a 24-hour period for $l(A_2)$ to $l(A_4)$, with y_2 to $y_4 = 0$ when θ_1 and attribute y_6 is included for the same reasons discussed in relation to previous decision contexts. These calculations or assignments are performed within a function, producing one 8 x 6 matrix and four 5 x 6 matrices. A subsequent function determines the multi-attribute utility of values in the matrices. All relevant software is presented in Appendix C.

H.4 Context: Vessel at Sea, Emergency

The expert decides between the last four deferral actions in Table H-5 which form the fourth decision tree. All calculation procedures and assignments follow the previous context.

Table H-6: Vessel at Sea Attribute Values

ATTRIBUTE y_a	$L(A_1)$	$l(A_2)$	$l(A_3)$	$l(A_4)$	$l(A_5)$
y_1	$Extra$	$(5\times Insp)+Extra$	$(4\times Insp)+Extra$	$(2\times Insp)+Extra$	$Insp+Extra$
y_2	$P(\theta_j \mid \vec{x})\times NewPump$	$P(\theta_j \mid \vec{x})\times NewPump$	$\frac{P(\theta_j \mid \vec{x})\times NewPump}{Correction^{1}}$	$\frac{P(\theta_j \mid \vec{x})\times NewPump}{Correction^{2}}$	$\frac{P(\theta_j \mid \vec{x})\times NewPump}{Correction^{4}}$
y_3	0	$P(\theta_j \mid \vec{x})\times P(LossVessel \mid \theta_j)\times NewVessel$	$\frac{P(\theta_j \mid \vec{x})\times P(LossVessel \mid \theta_j)\times NewVessel}{Correction^{1}}$	$\frac{P(\theta_j \mid \vec{x})\times P(LossVessel \mid \theta_j)\times NewVessel}{Correction^{2}}$	$\frac{P(\theta_j \mid \vec{x})\times P(LossVessel \mid \theta_j)\times NewVessel}{Correction^{4}}$
y_4	0	$P(\theta_j \mid \vec{x})\times P(Injury \mid \theta_j)\times NumCrew$	$P(\theta_j \mid \vec{x})\times P(Injury \mid \theta_j)\times NumCrew$	$P(\theta_j \mid \vec{x})\times P(Injury \mid \theta_j)\times NumCrew$	$P(\theta_j \mid \vec{x})\times P(Injury \mid \theta_j)\times NumCrew$
y_5	$y_1 \times EngRate$	$y_1 \times EngRate$	$\frac{y_1 \times EngRate}{Correction^{1}}$	$\frac{y_1 \times EngRate}{Correction^{2}}$	$\frac{y_1 \times EngRate}{Correction^{4}}$
y_6	1	1	1	1	1